

Wizcon® Supervisor™

La solution Internet complète de contrôle et d'information

WizPLC 3.0 CoDeSys

Manuel de développement



ELUTIONS Inc.
1300 East 8th Avenue
Suite 200
Tampa, FL 33605
USA
tel +1 (813) 371-5500
fax +1 (813) 371-5501

Wizcon Systems SAS
Parc Technologique de Lyon
12 allée Irène Joliot-Curie
F-69791 Saint-Priest Cedex
France
tel +33 (0)4 72 47 98 98
fax +33 (0)4 72 47 98 99

Wizcon Systems BV
Concordiaweg 149-151
Postbus 351
NL-4200 AJ Gorinchem
Nederland
tel +31 (0)183 646 303
fax +31 (0)183 621 601

Wizcon Systems Ltd
The Gate Hotel, Scotland Gate
Northumberland
NE62 5SS
UK
tel +44 (0)845 606-6120
fax +44 (0)845 606-6121

www.wizcon.com
http://support.wizcon.com

Manuel de développement de programmes pour automates programmables avec

CoDeSys 2.3

Toutes les mesures existantes ont été prises afin de garantir l'exactitude et l'intégralité de la documentation présente. Etant donné que des fautes restent toujours possible, malgré toutes les précautions qui sont prises, nous vous serions reconnaissants de bien vouloir nous faire part de vos remarques et de vos suggestions.

Editeur:

3S - Smart Software Solutions GmbH
Fischerstraße 19
D-87435 Kempten
Tél. +49/ 831/ 5 40 31 - 0
Fax +49/ 831/ 5 40 31 - 50

Edition: 02.06.2004
Document Version: 3.1, CoDeSys V2.3.3.3

Contenu

1 Bref aperçu de CoDeSys

1.1 Qu'est-ce que CoDeSys?

CoDeSys est un environnement de développement complet destiné à votre automate (**CoDeSys** signifie Controller Development System).

CoDeSys permet au programmeur d'AP d'aborder facilement les puissants outils de langage de la CEI. L'utilisation des éditeurs et des fonctions de débogage s'inspire d'environnements de développement perfectionnés des langages de programmation évolués (tels que Visual C++).

1.2 Aperçu des fonctionnalités de CoDeSys

Quelle est la structure d'un projet ?

Un projet est enregistré dans un fichier; le fichier reçoit le même nom que le projet. Le premier module à être créé dans un projet est automatiquement nommé PLC_PRG. C'est le point de départ de l'exécution (correspondant à la fonction "main" dans un programme C), à partir duquel d'autres blocs peuvent être appelés (programmes, blocs fonctionnels et fonctions).

Si vous avez défini une configuration de tâche, il n'est plus nécessaire de créer un programme nommé PLC_PRG. Pour plus de détails se reporter au chapitre 6.7, Configuration des tâches.

CoDeSys distingue différents types d'objets dans un projet : blocs, types de données, éléments de visualisation (visualisation) et ressources.

Vous trouvez la liste complète des objets relatifs à votre projet dans l'Organisateur d'objets.

Comment réaliser mon projet ?

Il vous faut d'abord configurer votre automate de façon à pouvoir vérifier l'exactitude des adresses utilisées dans le projet.

Ensuite, vous pouvez créer les blocs nécessaires à votre projet.

Enfin, vous pouvez programmer les blocs dont vous avez besoin dans les langages souhaités.

Après avoir terminé la programmation, vous pouvez compiler le projet et supprimer les erreurs indiquées, le cas échéant.

Comment tester mon projet ?

Après avoir supprimé toutes les erreurs, affichez la simulation, accédez à la simulation de l'automate programmable et chargez' votre projet dans l'automate. **CoDeSys** fonctionne à présent en ligne.

Vous pouvez désormais ouvrir la fenêtre de configuration de votre automate et vérifier le bon déroulement de votre projet. Pour ce faire, définissez manuellement les entrées et observez si les sorties sont modifiées comme prévu. En outre, vous pouvez observer l'évolution de la valeur des variables locales. Dans le gestionnaire d'espion et des recettes, vous pouvez configurer les jeux de données dont vous voulez observer les valeurs.

Débogage avec CoDeSys

Dans le cas d'erreurs de programmation, vous pouvez définir des points d'arrêt. Lorsque l'exécution s'interrompt à un tel point d'arrêt, vous pouvez visualiser les valeurs de toutes les variables du projet à cet instant. Avec une exécution pas à pas, vous pouvez vérifier si la logique de votre programme est correcte.

Fonctionnalités en ligne supplémentaires

Voici une autre fonction de débogage de **CoDeSys**: vous pouvez assigner une certaine valeur aux variables du programme ainsi qu'aux entrées et sorties.

Lors du mode En ligne, un **Journal** enregistre chronologiquement les processus et les actions de l'utilisateur ainsi que les processus internes.

Le contrôle du déroulement vous permet d'identifier les lignes de programme qui ont été exécutées. L'enregistrement de l'histogramme vous donne la possibilité d'enregistrer et de visualiser l'évolution des variables pendant un temps prolongé, lorsque le système fonctionne avec un temps de cycle non ralenti. Cette fonction doit être activée dans la configuration du système cible.

En fonction également de la configuration du système cible, vous disposez en option d'un **navigateur d'automate programmable** vous permettant de contrôler certaines informations provenant de l'automate.

Une fois le projet élaboré et testé, il reste à le charger dans le matériel et à le tester à nouveau dans le matériel. Vous disposez des mêmes fonctions en ligne que pour la simulation.

Possibilités supplémentaires offertes par CoDeSys

La totalité du projet peut être documentée ou exportée vers un fichier texte à tout moment.

CoDeSys dispose d'une interface des symboles, d'une interface DDE ainsi que d'une interface COM permettant la **communication**. Une passerelle ainsi qu'un serveur OPC et un serveur DDE font partie de l'installation standard de CoDeSys.

L'utilisation du jeu approprié de **Configuration de système cible**, pouvant être chargé par le biais d'un fichier cible (Target Support Package), permet d'appliquer le même projet CoDeSys dans différents systèmes cible.

Des variables globales réseau et un **Gestionnaire des paramètres** (selon la configuration du système cible) peuvent également être utilisés en option afin d'échanger des données avec d'autres automates au sein d'un réseau.

ENI: L'interface 'Engineering Interface' peut être utilisée afin d'accéder, par le biais de son propre serveur ENI, à une base de données externe dans laquelle des modules CoDeSys ou des fichiers de compilation sont gérés. Ainsi, ces derniers sont également accessibles à d'autres clients du serveur ENI, ce qui permet, par exemple, une exploitation par plusieurs utilisateurs lors de la création de projets CoDeSys, un pool commun de données pour d'autres outils outre CoDeSys, ainsi qu'une gestion de version.

Outils: Le mécanisme Outils sert à intégrer dans CoDeSys des fichiers exe spécifiques au système cible. On peut en outre définir des fichiers qui doivent être chargés sur l'automate programmable. On peut définir au préalable des liens vers des outils pour un système cible dans un fichier cible, ou insérer ces liens individuellement dans le projet via l'arborescence des ressources. La disponibilité de cette fonction Outils dépend du système cible.

Grâce à **CoDeSys HMI**, une visualisation CoDeSys peut être utilisée purement et simplement comme zone de commande, ou encore, selon le système cible, comme **Visualisation Web** et/ou **Visualisation Cible**. Ces dernières possibilités permettent d'appeler la visualisation reprenant les données de l'automate en cours, cela par le biais d'Internet ou encore de manière directe sur le moniteur de l'ordinateur de commande.

Attribution de **licences** pour bibliothèques: Les bibliothèques créées au sein de CoDeSys peuvent être munies d'informations relatives à la licence, ce qui rend leur utilisation liée à une licence.

SoftMotion: Cette fonction est une sorte de boîte à outils standard permettant de réaliser des mouvements - des mouvements simples à un seul axe aux mouvements complexes multi-dimensionnels, en passant par les mouvements de came, cela dans l'environnement de développement de CoDeSys. Toute la logique du programme est traitée dans le programme API, et seules les informations touchant exclusivement aux mouvements sont exécutées dans les modules de bibliothèque. SoftMotion nécessite une licence propre et doit être supportée par le système cible. (Voir documentation d'utilisateur séparée)

2 Qui fait quoi dans CoDeSys

2.1 Composantes d'un projet

Projet

Un projet contient tous les objets relatifs à un programme d'automatisation. Un projet est stocké dans un fichier auquel on attribue le nom du projet. Un projet comprend les objets suivants:

modules, types de données, visualisations, ressources et librairies.

Module

Les fonctions, les blocs fonctionnels et les programmes sont des modules pouvant être complétés par des actions.

Chaque module est composé d'une partie de déclaration et d'un corps. Le corps est écrit dans un des langages de programmation de la CEI, à savoir IL, ST, SFC, FBD LD ou CFC.

CoDeSys supporte tous les modules standard CEI. Pour utiliser ces modules dans votre projet, vous devez intégrer la bibliothèque standard.lib dans votre projet.

Les modules peuvent appeler d'autres modules. Les récursions ne sont toutefois pas autorisées.

Fonction

Une fonction est un module dont l'exécution produit un seul élément de donnée (pouvant éventuellement être composé de plusieurs éléments, comme par exemple des champs ou des structures) et qui peut apparaître sous forme d'opérateurs dans des expressions, s'il est appelé dans des langages littéraux.

En déclarant une fonction, il faut veiller à ce que la fonction reçoive un type, c.-à-d. qu'il faut entrer à la suite du nom de fonction un signe deux-points suivi d'un type.

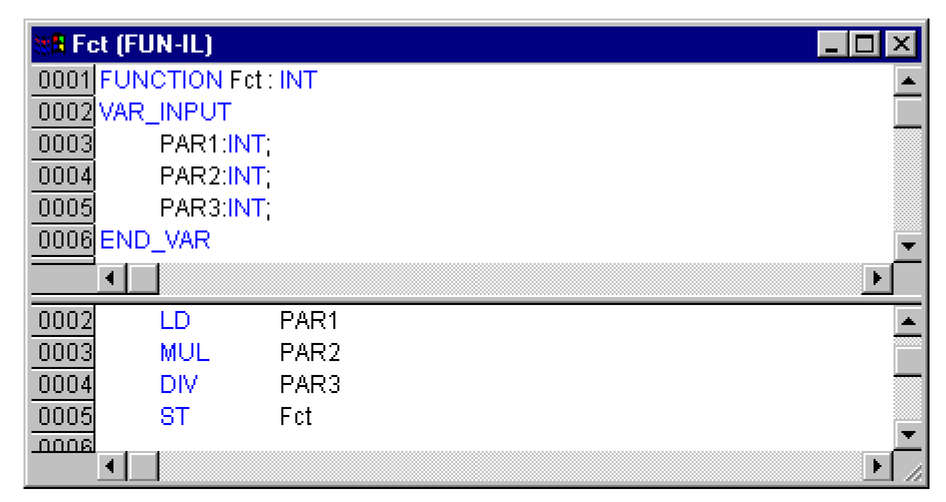
Une déclaration de fonction correcte ressemble à ceci:

```
FUNCTION Fct: INT
```

En outre, il faut affecter un résultat à la fonction, c.-à-d. que le nom de fonction est utilisé comme variable de sortie.

Une déclaration de fonction commence par le mot-clé FUNCTION.

Exemple d'une fonction en langage liste d'instructions



En langage ST, l'appel d'une fonction peut apparaître sous forme d'opérandes dans des expressions.

Les fonctions ne disposent pas d'états internes, c.-à-d. que plusieurs appels d'une même fonction en utilisant les mêmes arguments (paramètres d'entrée) fournissent toujours la même valeur (sortie).

Voici quelques exemples d'appels de la fonction:

en langage IL:

```
LD 7
Fct 2,4
ST Résultat
```

en langage FBD:

```
Résultat := Fct(7, 2, 4);
```

en langage FBD:

```
     Fct
7 -|PAR1|——Ergebnis
4 -|PAR2|
2 -|PAR3|
```

Dans les diagrammes fonctionnels en séquence (SFC) un appel de fonction ne peut être réalisé qu'à l'intérieur d'une étape ou d'une transition.

Remarque : Si dans votre projet vous définissez une fonction nommée CheckBounds, vous pourrez vérifier automatiquement les débordements de plage à l'intérieur de votre projet !

Remarque : Si dans votre projet vous définissez des fonctions nommées CheckDivByte, CheckDivWord, CheckDivDWord et CheckDivReal, vous pourrez vérifier la valeur du diviseur lors de l'utilisation de l'opérateur DIV, par exemple pour éviter une division par zéro.

Remarque : Si vous définissez les fonctions **CheckRangeSigned** et **CheckRangeUnsigned**, vous pouvez repérer automatiquement en mode En ligne les débordements de plage pour des variables qui sont déclarées à l'aide de types domaines partiels.

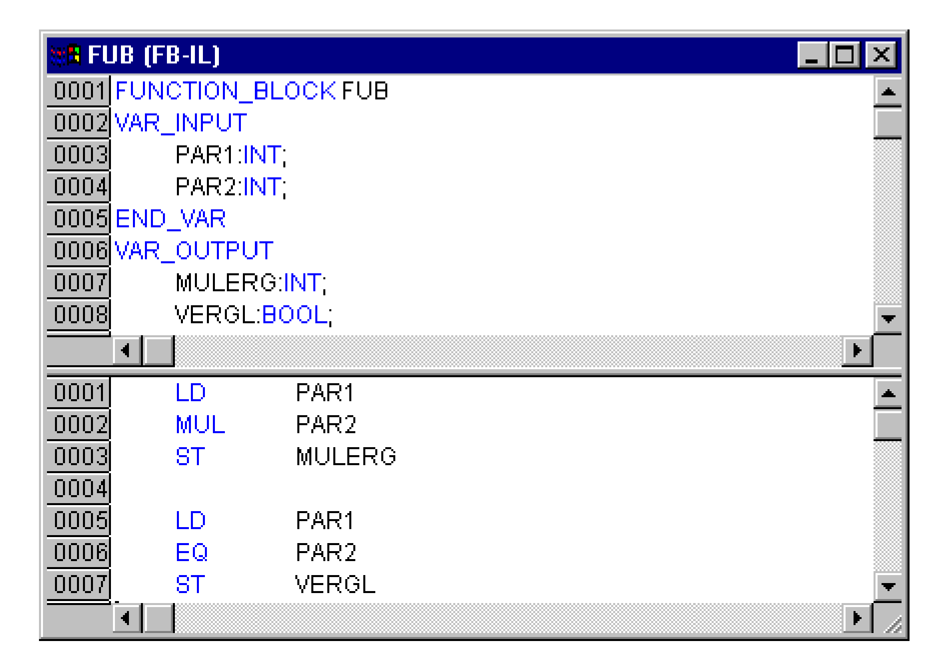
Les noms des fonctions citées sont réservés en raison des possibilités d'utilisation décrites ici.

Bloc fonctionnel

Un bloc fonctionnel est un module dont l'exécution produit une ou plusieurs valeurs. Contrairement à une fonction, un bloc fonctionnel ne produit pas de valeur renvoyée.

Une déclaration de bloc fonctionnel commence par le mot clé FUNCTION_BLOCK et se termine par le mot clé END_FUNCTION_BLOCK.

Voici un exemple de bloc fonctionnel à deux variables d'entrée et à deux variables de sortie, en langage IL. La première sortie est le produit des deux entrées tandis que la deuxième sortie est le résultat booléen de la comparaison à l'identique.



Instances de blocs fonctionnels

Il est possible de créer des reproductions d'un bloc fonctionnel, appelées *Instances* (copies).

Chaque instance possède un identificateur propre (le nom d'instance) et une structure de données, qui contient les entrées, les sorties et les variables internes relatives à cette instance. Les instances sont déclarées comme les variables de façon locale ou globale, en donnant comme type de l'identificateur le nom du bloc fonctionnel.

Voici, à titre d'exemple, une instance nommée INSTANZ, relative au bloc fonctionnel FUB:

```
INSTANZ: FUB;
```

Les blocs fonctionnels sont toujours appelés par l'intermédiaire des instances, telles que définies ci-dessus.

Seuls les paramètres d'entrée et de sortie sont accessibles du dehors d'une instance de bloc fonctionnel, non les variables internes du bloc fonctionnel.

Voici un exemple où l'on accède à une variable d'entrée:

Le bloc fonctionnel fb a une variable d'entrée nommée in1, de type int.

```
PROGRAM prog
VAR
 inst1:fb;
END_VAR
LD 17
ST  inst1.in1
CAL inst1
END_PROGRAM
```

Les parties de déclaration des blocs fonctionnels et des programmes peuvent contenir des déclaration d'instances. Les déclarations d'instances ne sont pas admises à l'intérieur de fonctions.

L'accès à une instance de bloc fonctionnel est limité au module, ce qui est bien le but poursuivi par la création de l'instance, sauf si l'instance a été déclarée de façon globale.

Le nom d'instance d'une instance de bloc fonctionnel peut être utilisé comme entrée de fonction ou de bloc fonctionnel.

Remarque : Entre deux exécutions successives d'un même bloc fonctionnel, toutes les valeurs sont conservées. C'est pourquoi un bloc fonctionnel appelé avec les mêmes arguments ne produit pas toujours les mêmes valeurs de sortie !

Remarque : Si le bloc fonctionnel comporte au moins une variable rémanente, l'entièreté de l'instance sera sauvegardée dans la zone de rémanence.

Appel d'un bloc fonctionnel

Vous pouvez accéder aux variables d'entrée et de sortie d'un bloc fonctionnel à partir d'un autre module en créant une instance du bloc fonctionnel et en indiquant la variable souhaitée au moyen de la syntaxe suivante :

<nom d'instance>.<nom de variable>

Affectation des paramètres à l'appel :

Si vous souhaitez définir les paramètres d'entrée et/ou de sortie lors de l'appel, vous pouvez le faire au moyen des langages littéraux IL et ST, cela en attribuant des valeurs aux paramètres suivant le nom d'instance du bloc fonctionnel donné entre parenthèses. L'affectation pour les paramètres d'entrée se fait par ":=" comme pour l'endroit de déclaration lors de l'initialisation de variables, et par "=>") dans le cas de paramètres de sortie.

Si l'instance est insérée dans la fenêtre d'implémentation d'un module ST ou IL, en utilisant la liste de sélection pour l'édition (<F2>) et l'option **Avec arguments**, cette instance sera automatiquement représentée avec ses paramètres dans cette syntaxe. Il n'est alors pas absolument nécessaire d'affecter les paramètres.

Exemple :

FBINST est une variable locale du type d'un bloc fonctionnel, comprenant la variable d'entrée xx et la variable de sortie yy. Lors de l'appel de FBINST via la liste de sélection pour l'édition, elle est insérée dans un programme ST sous la forme suivante: FBINST1(xx:= , yy=>);

Variables d'entrée / de sortie à l'appel :

Veuillez noter que **les variables d'entrée / de sortie (VAR_IN_OUT)** d'un bloc fonctionnel sont transmises comme **Pointeur**. Il est dès lors impossible de leur attribuer une constante lors de l'appel, et on ne peut y avoir accès de l'extérieur en lecture ou en écriture.

Exemple

Appel d'une variable VAR_IN_OUT inout1 du bloc fonctionnel fubo au sein d'un module ST:

```
VAR
 inst:fubo;
 var1:int;
END_VAR
var1:=2;
inst(inout1:=var1);
```

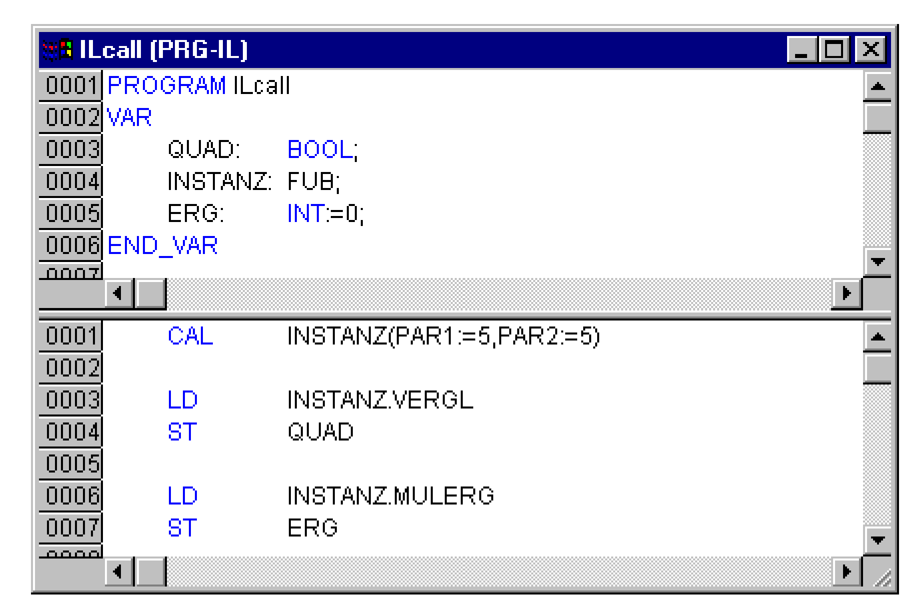
N'est pas autorisé : inst(inout1:=2); ou inst.inout1:=2;

Exemples d'appel du bloc fonctionnel FUB :

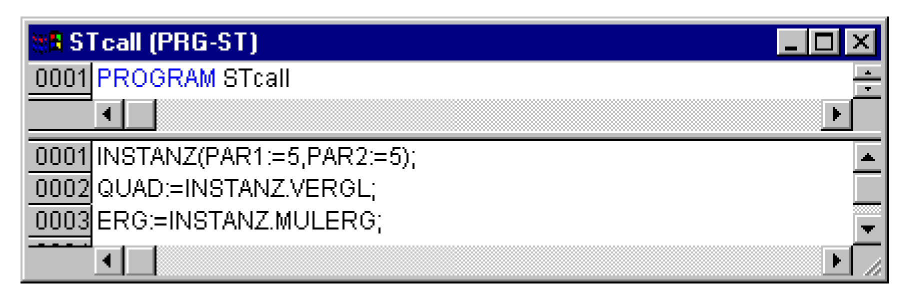
voir plus haut, paragraphe 'Bloc fonctionnel'

Le résultat de la multiplication est placé dans la variable ERG, le résultat de la comparaison est mémorisé dans QUAD. Supposons qu'une instance du bloc fonctionnel FUB soit déclarée sous le nom INSTANZ.

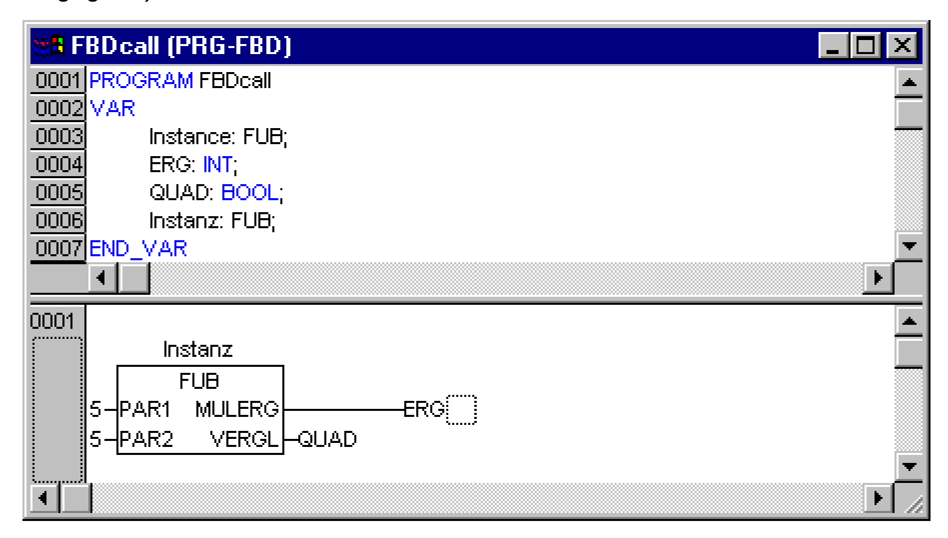
On appelle une instance d'un bloc fonctionnel en langage IL de la manière suivante :



On appelle une instance d'un bloc fonctionnel en langage ST (partie de déclaration comme en langage IL) de la manière suivante :



On appelle une instance d'un bloc fonctionnel en langage FBD (partie de déclaration comme en langage IL) de la manière suivante :

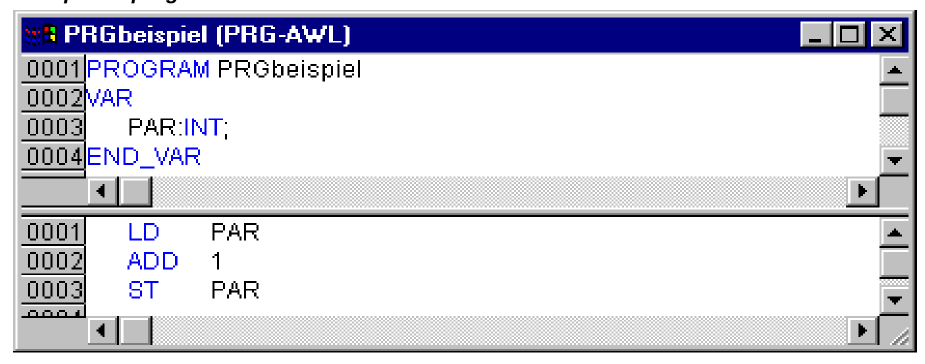


En SFC, les appels de blocs fonctionnels apparaissent seulement au niveau des étapes.

Programme

Un programme est un module qui produit lors de son exécution une ou plusieurs valeurs. Les programmes sont connus de manière globale dans l'ensemble du projet. Toutes les valeurs sont conservées entre deux exécutions successives d'un même programme.

Exemple de programme



Les programmes peuvent être appelés à partir de programmes et de blocs fonctionnels. Un appel de programme n'est pas autorisé dans une fonction. Il n'existe pas non plus d'instances de programmes.

Lorsqu'un programme est appelé à partir d'un module, et que cela entraîne des changements dans les valeurs du programme, ceux-ci sont conservés lors de l'appel suivant, même si le programme est appelé à partir d'un autre module.

Ceci est différent de l'appel d'un bloc fonctionnel. Dans ce cas, seules les valeurs de l'instance correspondant à ce bloc fonctionnel sont modifiées.

Si vous souhaitez définir les paramètres d'entrée et/ou de sortie - et donc définir les valeurs des variables d'entrée / de sortie - à l'appel d'un programme, il suffit, dans les langages IL et ST, d'attribuer des valeurs aux paramètres suivant le nom du programme donné entre parenthèses. L'affectation se fait par ":=" comme pour l'endroit de déclaration lors de l'initialisation de variables.

Si un programme est inséré dans la fenêtre d'implémentation d'un module ST ou IL, en utilisant la liste de sélection pour l'édition (<F2>) et l'option **Avec arguments**, il sera automatiquement représenté avec ses paramètres dans cette syntaxe. Il n'est alors pas absolument nécessaire d'affecter les paramètres.

Exemple d'appels de programmes :

Dans un programme PRGexample2, la variable d'entrée in_var et la variable de sortie out_var sont toutes deux déclarées de type INT. La variable erg déclarée localement est également de type INT :

En langage IL:

```
CAL PRGexample2
LD PRGexample2.out_var
ST ERG
ou par la affichage direct des paramètres (liste de sélection pour l'édition "Avec arguments", voir plus haut):
CAL PRGexample2(in_var:=33, out_var=>erg )
```

En langage ST:

```
PRGexample;
Erg := PRGexample2.out_var;
ou par la affichage direct des paramètres (liste de sélection pour l'édition "Avec arguments", voir plus haut):
PRGexample2(in_var:=33, out_var=>erg );
```

En langage FBD:

```
     PRGEXAMPLE 2
33 -|in_var   out_var|---erg
```

Exemple illustrant une possibilité de séquence d'appels de PLC_PRG:

Voir pour ce faire le programme PRGexample dans l'illustration au début de ce chapitre:

```
LD 0
```

```
ST PRGexample.PAR (*La valeur 0 est affecté par défaut à PAR*)
CAL ILcall (*ERG prend la valeur 1 dans ILcall*)
CAL STcall (*ERG prend la valeur 2 dans STcall*)
CAL FBDcall (*ERG prend la valeur 3 dans FBDcall*)
```

Si, à partir d'un programme principal, on initialise la variable PAR du programme PRGexample avec la valeur 0 et qu'ensuite des programmes sont appelés successivement au moyen des appels de programme ci-dessus, le résultat Erg à l'intérieur des programmes aura comme valeur 1, 2 et 3. Si on inverse l'ordre des appels, on observe le changement en conséquence des valeurs des paramètres de résultat correspondants.

PLC_PRG

PLC_PRG est un module prédéfini particulier. Chaque projet doit contenir ce programme particulier. Ce module est appelé exactement une fois par cycle de commande.

Lorsque 'Projet' 'Insérer objet' est exécuté pour la première fois après la création d'un nouveau projet**,** la préaffectation dans le dialogue des modules est un module nommé PLC_PRG de type programme. Nous vous conseillons de ne pas modifier cette préaffectation !

Si des tâches ont été définies, le projet ne peut contenir de programme PLC_PRG, car dans ce cas l'ordre d'exécution est fonction de l'affectation des tâches.

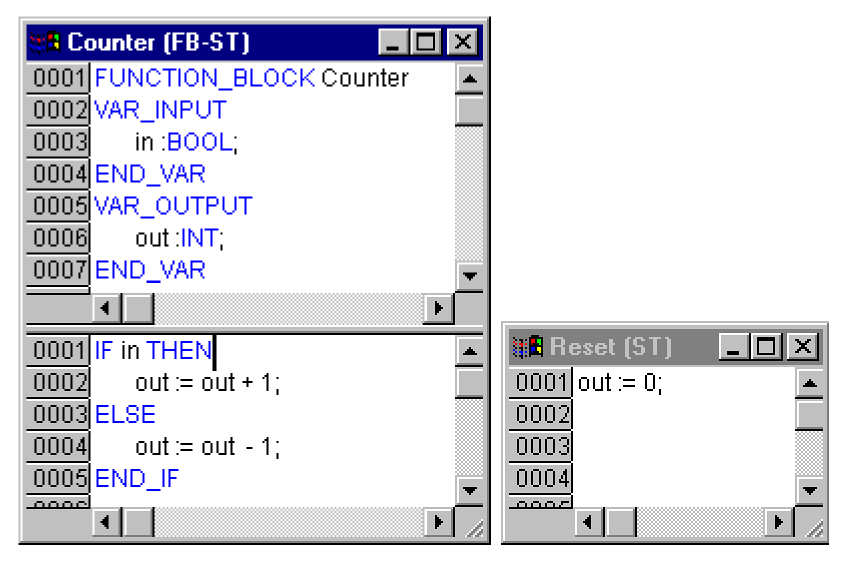
Attention : ne pas effacer le module PLC_PRG et ne pas modifier son nom (dans le cas où vous n'utilisez pas de configuration des tâches, voir chapitre 6.6.11, Configuration de l'automate en mode En Ligne !). Le programme PLC_PRG est toujours le programme principal dans un programme monotâche.

Action

Il est possible de définir des actions associées aux blocs fonctionnels et aux programmes. L'action constitue une implémentation supplémentaire, qui peut même être créée dans un langage autre que celui de l'implémentation "normale". Toute action est désignée par un nom.

Une action travaille avec les données du bloc fonctionnel ou du programme auquel elle est associée. L'action fait appel aux mêmes variables d'entrée / de sortie et aux mêmes variables locales que l'implémentation „normale".

Exemple d'une action d'un bloc fonctionnel



Dans l'exemple présent, si le bloc fonctionnel Counter est appelé, la variable de sortie out est incrémentée ou décrémentée en fonction de la variable d'entrée in. Si l'action Reset associée au bloc fonctionnel est appelée, la variable de sortie out est mise à zéro. Il s'agit de la même variable out dans les deux cas de figure.

<u>Appel d'une action :</u>

Une action est appelée au moyen de <Nom de programme>.<Nom d'action> ou <Nom d'instance>.<Nom d'action>. Veuillez observer la transcription en FBD (voir exemple ci-dessous) ! Si l'action doit être appelée à l'intérieur de son propre module, alors on utilise uniquement le nom d'action dans le cas des éditeurs littéraux et l'appel de bloc fonctionnel sans spécification d'instance dans le cas des éditeurs graphiques.

Exemples d'appel de l'action ci-dessus :

Déclaration pour tous les exemples:

```
PROGRAM PLC_PRG
VAR
 Inst : Counter;
END_VAR
```

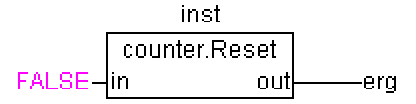
Appel de l'action 'Reset' dans un autre module programmé en **langage IL**:

```
CAL Inst.Reset(In := FALSE)
LD Inst.out
ST ERG
```

Appel dans un autre module programmé en **langage ST**:

```
Inst.Reset(In := FALSE);
Erg := Inst.out;
```

Appel dans un autre module programmé en **langage FBD**:



Remarques : Dans le cas de modules en diagramme fonctionnel en séquence (SFC), les actions jouent un rôle spécifique.
La norme CEI ne reconnaît aucune action, sauf dans le cas du diagramme fonctionnel en séquence (SFC).

Ressources

Vous avez besoin de ressources pour configurer et organiser votre projet et pour suivre l'évolution des valeurs des variables. Il s'agit :

- de Variables globales, qui peuvent être utilisées dans l'ensemble du projet ou du réseau
- la Configuration de l'alarme pour configurer des classes et groupes d'alarme, qui par example peuvent visualisée avec la visualisation de CoDeSys
- la Gestionnaire de bibliothèques pour l'administration des bibliotheques dans un projet
- de la Configuration de l'automate , pour configurer votre matériel
- de la Configuration des tâches, pour commander votre programme au moyen de tâches
- du Journal pour enregistrer les actions qui ont lieu lors d'une session
- du Gestionnaire d'espion et des recettes, pour visualiser des valeurs de variables et préaffecter des valeurs aux variables.
- de Configuration de la cible pour le choix et éventuellement la configuration définitive du système cible
- Environnement de travail avec représentation des options du projet
- Dependant de la configuration de la cible en outre ce sont disponibles les ressources suivantes:
- L'enregistrement des Histogramme, pour l'enregistrement graphique des valeurs des variables
- Le Manager des paramètres pour l'échange de données avec d'autres automates au sein d'un réseau

- PLC-Browser pour le contrôle de l'automate
- D'Outils par attacher des tools externes
- SoftMotion

Se reporter au chapitre 6, 'Ressources'.

Bibliothèques

Dans votre projet, vous pouvez intégrer une série de bibliothèques et utiliser les modules, les types de données et les variables globales correspondantes au même titre que les éléments définis directement dans le projet. La bibliothèques 'standard.lib' et util.lib sont mises à votre disposition par défaut.

Se reporter au chapitre 7, 'Gestion des bibliothèques'.

Types de données

En plus des types de données standards, l'utilisateur peut définir ses propres types de données. Il est possible de créer des structures, des types énumératifs et des références.

Se reporter pour cela à l'annexe, à 'Types de données standard' et à 'Types de données définis'.

Visualisation

CoDeSys peut vous fournir une représentation concrète des variables de votre projet au moyen d'une visualisation. A l'aide de la visualisation vous pouvez tracer en mode Hors Ligne des éléments géométriques. Lors de l'utilisation en mode En Ligne, ces éléments peuvent ensuite changer de forme / couleur / affichage du texte, en fonction de valeurs de variables déterminées.

Une visualisation peut également servir de zone de commande exclusive pour un projet avec CoDeSys HMI, ou encore sous forme de Visualisation Web ou de Visualisation Cible via Internet ou sur le système cible.

Se reporter pour cela au manuel 'CoDeSys Visualisation'.

2.2 Les Langages

2.2.1 Liste d'Instructions (IL)

La Liste d'Instructions (IL) est constituée d'une succession d'instructions. Avec chaque instruction commence une nouvelle ligne, qui contient un opérateur et, selon le type d'opération, un ou plusieurs opérandes séparés entre eux par des virgules.

Une *Etiquette* d'identification suivi de deux-points (:) peut être placée avant une instruction.

Lorsqu'une ligne contient un commentaire celui-ci doit obligatoirement constituer le dernier élément de la ligne. Des lignes vides peuvent être insérées entre les instructions.

Exemple :

```
LD 17
ST lint (* Kommentar *)
GE 5
JMPC next
LD idword
EQ istruct.sdword
STN test
next:
```

plus:

Modificateurs et opérandes en langage IL

Modificateurs et opérandes en langage IL

En langage IL, il est possible d'utiliser les opérateurs et modificateurs suivants:

Modificateurs:

- C pour JMP, CAL, RET: l'instruction est exécutée uniquement si l'expression précédente fournit le résultat TRUE.
- N pour JMPC, CALC, RETC: l'instruction est exécutée uniquement si l'expression précédente fournit le résultat FALSE.
- N autres: négation de l'opérande (mais pas de l'accumulateur)

Vous trouvez ci-dessous un tableau de tous les opérateurs en langage IL et leurs modificateurs autorisés, avec une description de ces opérateurs:

Opérateur	Modificateurs	Description
LD	N	Rendre le résultat courant égal à l'opérande
ST	N	Mémoriser le résultat courant à l'emplacement de l'opérande
S		Positionner l'opérande booléenne sur TRUE si et seulement si le résultat courant est TRUE
R		Remettre l'opérande booléenne sur FALSE si et seulement si le résultat courant est TRUE
AND	N,(	AND bit à bit
OR	N,(	OR bit à bit
XOR	N,(	OR exclusif bit à bit
ADD	(	Addition
SUB	(	Soustraction
MUL	(	Multiplication
DIV	(	Division
GT	(	>
GE	(	>=
EQ	(	=
NE	(	<>
LE	(	<=
LT	(	<
JMP	CN	Saut vers l'étiquette
CAL	CN	Appel d'un bloc fonctionnel
RET	CN	Retour d'un bloc fonctionnel appelé
)		Evaluation d'une opération différée

Vous trouvez une liste de tous les opérateurs CEI en annexe.

Exemple de programme en langage IL utilisant certains modificateurs:

```
LD      TRUE    (*charger TRUE dans l'accumulateur*)
ANDN    BOOL1   (*exécuter AND avec la négation de la valeur de la variable BOOL1*)
JMPC    label   (*si le résultat est TRUE, sauter à l'étiquette "label"*)

LDN     BOOL2   (*mémoriser la négation de la valeur de*)
ST      ERG     (*BOOL2 dans ERG*)
label:
LD      BOOL2   (*mémoriser la valeur de*)
ST      ERG     (*BOOL2 dans ERG*)
```

En langage IL, il est également possible de placer des parenthèses après une opération. La valeur entre parenthèses est alors considérée comme une opérande.

Par Exemple :

```
LD    2
MUL   2
ADD   3
ST    Erg
```

Dans ce cas la valeur de Erg est 7. En revanche, si l'on place des parenthèses:

```
LD    2
MUL   (2
ADD   3
)
ST    Erg
```

Dans ce cas cela donne 10 comme résultat de Erg, étant donné que l'opération MUL est exécutée seulement à partir du moment où l'on rencontre ")"; le calcul de l'opérande pour MUL donne alors 5.

Littéral Structuré (ST)

Le Littéral Structuré est constitué d'une série d'instructions, qui peuvent être exécutées en ayant défini des conditions (IF..THEN..ELSE) ou sous forme de boucles (WHILE..DO), comme c'est le cas pour les langages évolués.

Exemple :

```
IF value < 7 THEN
 WHILE value < 8 DO
  value := value + 1;
 END_WHILE;
END_IF;
```

Expressions

Une *expression* est une construction syntaxique qui, lorsqu'elle est évaluée, fournit une valeur.

Les expressions sont composées d'opérateurs et d'opérandes. Un opérande peut être soit une constante, soit une variable, soit un appel de fonction, soit une autre expression.

Evaluation des expressions

Une expression est évaluée par l'application des opérateurs en suivant des *règles de priorité* déterminées. L'opérateur possédant la plus forte priorité est appliqué en premier, puis c'est le tour des autres opérateur par ordre de priorité décroissante, jusqu'à ce que tous les opérateurs aient été appliqués.

Si des opérateurs ont même rang de priorité, ils sont appliqués successivement, en commençant à gauche et en se déplaçant vers la droite.

Vous trouvez ci-après un tableau avec les opérateurs du langage ST rangés par ordre de priorité décroissante:

Operation	**Symbole**	**Priorité**
Mise entre parenthèses	(Expression)	Priorité la plus élevée
Evaluation de fonction	Nom de fonction (liste d'arguments)	
Exponentiation	EXPT	
Négation Complément	- NOT	
Multiplication Division Modulo	* / MOD	
Addition Soustraction	+ -	
Comparaison	<,>,<=,>=	
Similitude Dissimilitudet	= <>	
AND booléen	AND	
OR exclusif booléen	XOR	
OR booléen	OR	Priorité la plus faible

En langage ST, les énoncés existants sont ordonnés dans un tableau et illustrés par des exemples:

Type d'énoncé	**Exemple**
Affectation	A:=B; CV := CV + 1; C:=SIN(X);
Lancement d'un bloc fonctionnel et utilisation de sortie FB	CMD_TMR(IN := %IX5, PT := 300); A:=CMD_TMR.Q
RETURN	RETURN;
IF	D:=B*B; IF D<0.0 THEN C:=A; ELSIF D=0.0 THEN C:=B; ELSE C:=D; END_IF;
CASE	CASE INT1 OF

	1: BOOL1 := TRUE; 2: BOOL2 := TRUE; ELSE BOOL1 := FALSE; BOOL2 := FALSE; END_CASE;
FOR	J:=101; FOR I:=1 TO 100 BY 2 DO IF ARR[I] = 70 THEN J:=I; EXIT; END_IF; END_FOR;
WHILE	J:=1; WHILE J<= 100 AND ARR[J] <> 70 DO J:=J+2; END_WHILE;
REPEAT	J:=-1; REPEAT J:=J+2; UNTIL J= 101 OR ARR[J] = 70 END_REPEAT;
EXIT	EXIT;
Enoncé Vide	;

Enoncés du langage Littéral structuré

Comme son nom l'indique, le Littéral Structuré est conçu pour permettre une programmation structurée. Cela signifie que le langage ST offre des structures prédéfinies pour programmer des constructions syntaxiques fréquemment utilisées, telles que des boucles.

Cela présente deux avantages, à savoir une probabilité d'erreurs moindre et une présentation plus lisible du programme.

Comparons par exemple deux séquences de programme identiques, programmées dans les langages IL et ST.

Il s'agit d'une boucle pour calculer des puissances de deux en **IL**:

bou Aperçucle:

```
LD      compteur
EQ      0
JMPC    fin
LD      Var1
MUL     2
ST      Var1
LD      compteur
SUB     1
ST      compteur
JMP     boucle
```

fin:

```
LD      Var1
ST      res
```

La même boucle programmée en langage **ST** donnerait ceci:

```
WHILE compteur<>0 DO
 Var1:=Var1*2;
 compteur:=compteur-1;
END_WHILE
res:=Var1;
```

Comme on peut le voir, la boucle programmée en langage ST est non seulement plus courte, mais aussi plus facile à lire, surtout si on imagine le cas de boucles emboîtées dans des constructions syntaxiques plus importantes.

Voici la signification des différentes structures existant en langage ST:

Opérateur d'affectation

Dans la partie gauche de l'affectation se trouve un opérande (variable, adresse), auquel est affecté, au moyen de l'opérateur :=, la valeur de l'expression située dans la partie droite de l'affectation.

Exemple :

```
Var1 := Var2 * 10;
```

Après avoir exécuté cette ligne, Var1 a une valeur dix fois plus grande que celle de Var2.

Appel de blocs fonctionnels en langage ST

Pour appeler un bloc fonctionnel en langage ST, il faut écrire le nom d'instance du bloc fonctionnel et puis affecter aux paramètres les valeurs souhaitées, entre parenthèses. Dans l'exemple suivant un temporisateur est appelé et des valeurs sont affectées aux paramètres IN et PT. Ensuite, la variable de résultat Q est affectée à la variable A.

On accède à la variable de résultat comme en langage IL à l'aide du nom du bloc fonctionnel suivi d'un point et du nom de la variable:

```
CMD_TMR(IN := %IX5, PT := 300);
A:=CMD_TMR.Q
```

Enoncé RETURN

L'énoncé RETURN peut être utilisé pour quitter une fonction, en définissant par exemple une condition pour cela.

Enoncé CASE

L'énoncé CASE permet de regrouper plusieurs énoncés liés à des conditions et qui possèdent la même variable de condition dans une construction syntaxique.

Syntaxe:

```
CASE <Var1> OF
<valeur_1>: <énoncé_1>
<valeur_2>: <énoncé_2>
...
<valeur_n>: <énoncé_n>
ELSE <énoncé_ELSE>
END_CASE;
```

Un énoncé CASE fonctionne suivant le schéma ci-dessous:

- si la variable <Var1> a comme valeur <valeur_i>, alors l'énoncé <énoncé_i> est exécuté;
- si <Var1> a une valeur différente des valeurs déclarées, alors l'énoncé <énoncé_ELSE> est exécuté;
- lorsque le même énoncé doit être exécuté pour différentes valeurs de la variable de condition, il est possible d'écrire ces valeurs à la suite, en les séparant par des virgules, ce qui revient à définir de façon commune les conditions pour l'exécution de l'énoncé.

Exemple :

```
CASE INT1 OF
1, 5: BOOL1 := TRUE;
 BOOL3 := FALSE;
2: BOOL2 := FALSE;
 BOOL3 := TRUE;
ELSE
 BOOL1 := NOT BOOL1;
 BOOL2 := BOOL1 OR BOOL2;
END_CASE;
```

Enoncé IF

L'énoncé IF permet de vérifier une condition et de définir cette condition pour l'exécution d'un énoncé.

Syntaxe:

IF <expression_booléenne_1> **THEN**

  `<énoncés_IF>`

{ELSIF <expression_booléenne_2> **THEN**

  `<énoncés_ELSIF_1>`

  .

  .

ELSIF <expression_booléenne_n> **THEN**

  `<énoncés_ELSIF_n-1>`

ELSE

  `<énoncés_ELSE>`**}**

END_IF;

La partie entre accolades {} est facultative.

Lorsque <expression_booléenne_1> donne comme résultat TRUE, seules les <énoncés_IF> sont exécutées, à l'exclusion de toutes les autres.

Dans le cas contraire, les expressions booléennes suivantes sont successivement vérifiées en commençant avec <expression_booléenne_2>, jusqu'à ce qu'une des expressions donne comme résultat TRUE. Ensuite, seuls sont exécutés les énoncés situés après cette expression booléenne et avant le ELSE ou le ELSEIF suivant.

Lorsque aucune des expressions booléennes ne donne comme résultat TRUE, alors seuls les <énoncés_ELSE> sont exécutés.

Exemple :

```
IF temp<17
THEN chauffage_on := TRUE;
ELSE chauffage_off := FALSE;
END_IF;
```

Dans ce cas, le chauffage est mis en marche si et seulement si la température descend en dessous de 17 degrés.

Enoncé d'itération FOR

L'énoncé d'itération FOR permet de programmer des processus répétés.

Syntaxe:

INT_Var :INT;

FOR <INT_Var> := <INIT_VAL> **TO** <END_VAL> **{BY** <pas>**} DO**

<énoncés>

END_FOR;

La partie entre accolades {} est facultative.

Les <énoncés> sont répétés aussi longtemps que le compteur <INT_Var> ne dépasse pas la valeur <END_VAL>. La vérification de cette condition est effectuée avant l'exécution des <énoncés>, de sorte que ceux-ci ne sont jamais exécutés lorsque la valeur <INIT_VAL> est supérieure à la valeur <END_VAL>.

A chaque fois que les <énoncés> ont été exécutés, <INT_Var> est incrémenté de la valeur du <pas>. Le pas peut avoir comme valeur n'importe quel nombre entier. On lui affecte par défaut la valeur 1. La boucle doit obligatoirement se terminer, étant donné que <INT_Var> ne peut qu'augmenter.

Exemple :

```
FOR compteur:=1 TO 5 BY 1 DO
Var1:=Var1*2;
END_FOR;
Erg:=Var1;
```

En supposant que la valeur 1 a été préaffectée à la variable Var1, celle-ci aura comme valeur 32 après la boucle FOR.

Remarque : <END_VAL> ne peut pas être égale à la valeur limite du type de variable du compteur <INT_Var>. P.ex. lorsque la variable compteur est du type SINT, <END_VAL> ne peut pas être 127 sinon il y a une boucle infinie.

Enoncé d'itération WHILE

L'énoncé d'itération WHILE peut être utilisé au même titre que l'énoncé d'itération FOR, à une différence près: la condition de terminaison peut être n'importe quelle expression booléenne. Cela signifie que l'on spécifie une condition qui, si elle est vérifiée, a pour effet l'exécution de la boucle.

Syntaxe:

```
WHILE <expression_booléenne> DO
 <énoncés>
END_WHILE;
```

Les <énoncés> sont exécutés de façon répétée aussi longtemps que <expression_booléenne> donne comme résultat TRUE. Si <expression_booléenne> donne comme résultat FALSE dès la première évaluation, alors les <énoncés> ne sont jamais exécutés. Si <expression_booléenne> ne prend jamais la valeur FALSE, alors les <énoncés> sont répétés indéfiniment, ce qui entraîne une erreur d'exécution.

Remarque : le programmeur doit veiller lui même à ce qu'aucune boucle infinie ne soit générée, en modifiant la condition de terminaison dans la partie <énoncés> de la boucle, par exemple en incrémentant ou en décrémentant un compteur.

Exemple :

```
WHILE compteur<>0 DO
 Var1 := Var1*2;
 compteur := compteur-1;
END_WHILE
```

Les énoncés WHILE et REPEAT sont dans un certain sens plus puissants que l'énoncé FOR, car il n'est pas nécessaire de savoir combien de fois la boucle sera parcourue avant son exécution. Dans certains cas, on est donc obligé de travailler uniquement avec ces deux types de boucle. Cependant, si l'on sait combien de fois la boucle sera parcourue, il est préférable de recourir à un énoncé FOR, car ce type de boucle exclut automatiquement les boucles infinies.

Enoncé d'itération REPEAT

L'énoncé d'itération REPEAT se distingue de l'énoncé d'itération WHILE par le fait que la vérification de la condition de terminaison est effectuée après que la boucle est exécutée. Il en résulte que la boucle est parcourue au moins une fois, quelle que soit la condition de terminaison.

Syntaxe:

```
REPEAT
 <énoncés>
UNTIL <expression_booléenne>
END_REPEAT;
```

Les <énoncés> sont exécutés aussi longtemps que <expression_booléenne> donne comme résultat TRUE.

Si <expression_booléenne> donne comme résultat TRUE dès la première évaluation, alors les <énoncés> sont exécutés exactement une fois. Si <expression_booléenne> ne prend jamais la valeur TRUE, alors les <énoncés> sont répétés indéfiniment, ce qui entraîne une erreur d'exécution.

Remarque : le programmeur doit veiller lui même à ce qu'aucune boucle infinie ne soit générée, en modifiant la condition de terminaison dans la partie <énoncés> de la boucle, par exemple en incrémentant ou en décrémentant un compteur.

Exemple :

```
REPEAT
 Var1 := Var1*2;
 compteur := compteur-1;
UNTIL
 compteur=0
END_REPEAT
```

Enoncé EXIT

Si l'énoncé EXIT apparaît dans une boucle FOR, WHILE ou REPEAT, alors la boucle la plus intérieure est terminée, indépendamment de la condition de terminaison.

2.2.2 Diagramme Fonctionnel en Séquence (SFC)

Le diagramme fonctionnel en séquence (SFC) est un langage graphique, qui permet de décrire la succession chronologique d'actions dans un programme. On utilise pour ce faire des éléments d'étape auxquels certaines actions sont affectées et dont la suite est commandée par des éléments de transition.

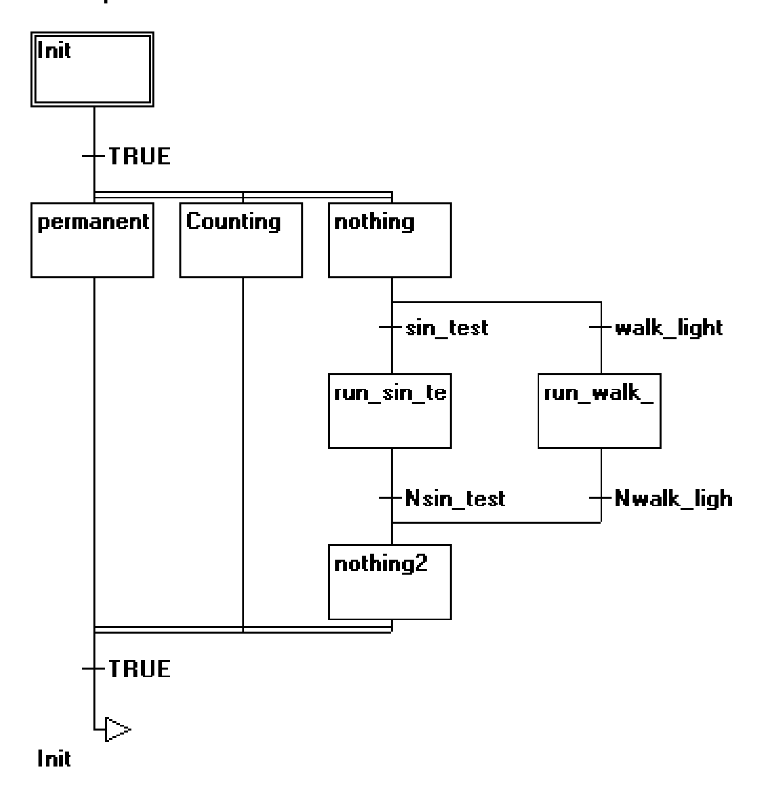
Etape

Un module écrit en SFC est constitué d'une succession d'étapes, qui sont reliées entre elles par des liaisons dirigées (transitions).

Il existe deux types d'étapes.

- La forme simplifiée est constituée d'une action et d'un drapeau, qui indique si l'étape est active ou inactive. Pour chaque étape dont une action est implémentée, un petit triangle apparaît dans le coin supérieur droit de l'étape.
- Une étape CEI est constituée d'un drapeau et d'une ou de plusieurs actions attribuées à l'étape. Les actions associées apparaissent à droite de l'étape. Des informations supplémentaires sont fournies par la suite.

Example d'un Réseau en SFC:



Action

Une action peut se présenter soit comme un ensemble d'instructions en langage IL, soit comme un ensemble d'énoncés en langage ST, soit comme un ensemble de réseaux en langage FBD, soit comme une ensemble d'échelons en langage LD, ou contenir un diagramme fonctionnel en séquence (SFC).

Dans le cas des étapes simplifiées, une action est toujours associée à son étape. Pour éditer une action, double-cliquez avec votre souris sur l'étape à laquelle est associée l'action, ou bien marquez l'étape et exécutez dans la commande de menu "**Zoom action/transition**" dans le menu 'Extras'. En outre, une action d'entrée et/ou de sortie par étape est possible.

Les actions des étapes CEI sont attachées dans l'Organisateur d'objets directement sous le module SFC correspondant et sont chargés dans l'éditeur par un double-clic ou par l'activation de la touche <Entrée>. De nouvelles actions peuvent être crées au moyen de '"rojet' + "Ajouter une action".

Action d'entrée et de sortie

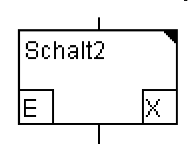
Il est possible d'ajouter à une étape, en plus de l'action d'étape, une action d'entrée et une action de sortie. Une action d'entrée est exécutée une seule fois, immédiatement après que l'étape ait été activée. Une action de sortie est exécutée une seule fois, avant que l'étape ne soit désactivée.

Une étape avec action d'entrée est indiquée par un 'E' dans le coin inférieur gauche, alors qu'une étape avec action de sortie est signalée par un 'X' dans le coin inférieur droit.

Les actions d'entrée et de sortie peuvent être implémentées dans le langage de votre choix. Pour éditer une action d'entrée ou de sortie, double-cliquez sur le coin correspondant de l'étape.

Les actions d'entrée et de sortie peuvent être définies uniquement pour une étape simplifiée et non pour une étape CEI

Voici un exemple d'étape avec action d'entrée et de sortie:



Transition / condition de transition

Entre les étapes se trouvent les "transitions".

Une condition de transition doit être avoir la valeur TRUE ou FALSE. Ainsi, elle est soit une variable booléenne, une adresse booléenne ou une constante booléenne. Elle peut également comprendre une suite d'instructions avec résultat booléen en syntaxe ST (p.ex.(i <= 100) AND b) ou un autre langage au choix (voir ‘Extras' 'Action Zoom/Transition'). Une transition ne peut cependant contenir un programme, un bloc fonctionnel ou des affectations !

Pour l'analyse des expressions de transition, on peut définir le drapeau *SFCErrorAnalyzationTable*.

Remarque : Outre les transitions, vous pouvez également utiliser le mode par étapes pour passer à l'étape suivante, voir à cet effet SFCtip et SFCtipmode.

Etape active

Après l'appel du module SFC, l'action correspondant à l'étape initiale (entourée en double) sera ensuite exécutée. Une étape dont l'action sera exécutée, est appelée active. En mode En Ligne, les étapes actives sont représentées en bleu.

A chaque étape correspond un drapeau, lequel mémorise l'état de l'étape. Le drapeau d'étape (état actif ou inactif de l'étape) sera représenté par la valeur logique d'un élément booléen <nom_du_pas>.x. Cette variable booléenne a la valeur TRUE (vraie) quand l'étape correspondante est active et FALSE (fausse) quand elle est inactive. Cette variable est déclarée implicitement et peut être utilisée dans toutes les actions et transitions du module SFC.

Dans un cycle de commande, toutes les actions appartenant à des étapes actives sont exécutées. Ensuite, les étapes qui suivent les étapes actives deviennent actives quand les conditions de transition des étapes suivantes sont TRUE. Les étapes à présent actives ne sont exécutées qu'au prochain cycle.

Remarque : Si l'étape active contient une action de sortie, celle-ci ne sera exécutée qu'au prochain cycle, pour autant que la transition suivante ait la valeur TRUE.

Etape CEI

A côté des étapes simplifiées, les étapes conformes aux normes CEI peuvent être utilisées dans le langage SFC.

Pour pouvoir utiliser des étapes CEI, vous devez intégrer dans votre projet une bibliothèque SFC particulière nommée **Iecsfc.lib**.

Un maximum de neuf actions peut être affecté à une étape CEI. Les actions des étapes CEI ne sont pas rattachées de manière fixe à une étape, comme par exemple les actions d'entrée, d'étape ou de sortie pour les étapes simplifiées, mais existent séparément des étapes et peuvent être utilisées plusieurs fois à l'intérieur de votre module. Pour cela, elles doivent être associées aux étapes souhaitées avec la commande "Relier action" dans le menu 'Extras'.

Outre les actions, des variables booléennes peuvent aussi être affectées aux étapes.

En fonction du qualificateur (qualifier), les actions et variables booléennes sont activées ou désactivées, parfois avec temporisation. Comme une action peut rester active, même si l'étape suivante est déjà terminée, par ex. avec le qualificateur S (set), des déroulements parallèles peuvent être obtenus.

Une variable booléenne associée est positionnée ou remise à zéro à chaque appel du module SFC. En clair, la valeur TRUE ou FALSE lui est à chaque fois attribuée.

Les actions associées à une étape CEI sont représentées par une case se trouvant à la droite de l'étape et divisée en deux. Le champ de gauche contient le qualificateur, éventuellement avec une constante de temps et celui de droite le nom de l'action.

Exemple d'une étape CEI avec deux actions :

Step1	R	Licht1
	S	Licht2

Pour pouvoir suivre facilement le processus, toutes les actions actives en mode En Ligne sont représentées en bleu comme les étapes actives. Après chaque cycle, on vérifie quelles actions sont actives.

Observez également à cet effet les limitations lors de l'utilisation de qualificatifs temporels pour des actions utilisées plusieurs fois au cours d'un même cycle (voir 'Qualificatifs') !

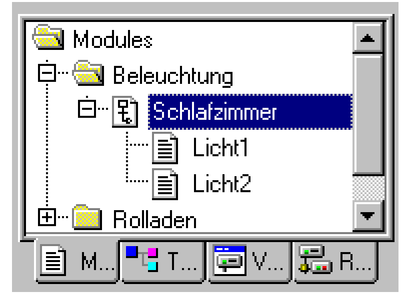
Remarque : Si une action est désactivée, elle sera exécutée **encore une fois**. Ceci signifie que chaque action est exécutée au moins deux fois (même une action avec le qualificatif P).

Lors d'un appel, les actions désactivées sont exécutées en premier lieu dans l'ordre alphabétique, puis les actions activées, elles aussi dans l'ordre alphabétique.

Le fait qu'une étape nouvellement introduite soit ou non une étape CEI dépend de la sélection de la commande "Utiliser les étapes CEI" dans le menu 'Extras'.

Dans l'Organisateur d'objets, les actions sont attachées directement sous le module SFC correspondant. De nouvelles actions peuvent être crées au moyen de la commande "Ajouter une action" du menu 'Projet'.

Module SFC avec actions dans l'Organisateur d'objets:



Qualificatif

Pour associer des actions à des étapes CEI, vous disposez des qualificatifs (qualifier) suivant:

N	Non-stored (non mémorisé)	l'action demeure active aussi longtemps que l'étape demeure active
R	overriding Reset (remise à zéro prioritaire)	désactivation de l'action
S	Set (Stored) (positionné (mémorisé))	activation de l'action, qui demeure ensuite active jusqu'au prochain reset
L	time Limited (limite dans le temps)	activation de l'action pendant une durée déterminée
D	time Delayed (temporisé)	activation de l'action après un certain temps, pour autant que l'étape demeure active
P	Pulse (impulsion)	l'action est exécutée exactement une fois, lorsque l'étape est activée
SD	Stored and time Delayed	activation de l'action après un certain temps; l'action demeure ensuite active jusqu'au prochain reset

	(mémorisé et temporisé)	
DS	Delayed and Stored (temporisé et mémorisé)	activation de l'action après un certain temps, pour autant que l'étape demeure active; l'action demeure alors active jusqu'au prochain reset
SL	Stored and time Limited (mémorisé et limité dans le temps)	activation de l'action pendant une durée déterminée

Les qualificatifs L, D, SD, DS et SL doivent être accompagnés d'une spécification de temps sous format de constante TIME, p.ex. L T#5s.

Attention : Si la même action avec qualificatifs influant sur le déroulement temporel est utilisée dans deux étapes se succédant directement, le qualificatif temporel ne peut plus être efficace lors de la deuxième utilisation. Pour contourner ce problème, il suffit d'introduire une étape intermédiaire qui verrait, en plus du cycle à effectuer, une réinitialisation de l'état d'action.

Variables implicites en SFC

Il existe des variables déclarées implicitement en SFC qui peuvent être utilisées.

A chaque étape correspond un drapeau, lequel mémorise l'état de l'étape. Le drapeau d'étape (état actif ou inactif de l'étape) s'appelle **<StepName>.x** dans le cas des étapes CEI et **<StepName>** dans le cas des étapes simplifiées. Cette variable booléenne a la valeur TRUE (vraie) quand l'étape correspondante est active et FALSE (fausse) quand elle est inactive. Elle peut être utilisée dans chaque action et transition d'un module SFC.

L'état actif ou inactif d'une action CEI peut être contrôlé à l'aide de la variable **<ActionName>.x**.

Dans le cas des étapes CEI, les variables implicites **<StepName>.t** peuvent être utilisées pour contrôler la durée des étapes.

Vous pouvez également accéder aux variables implicites à partir d'autres programmes. Exemple : boolvar1:=sfc1.step1.x; step1.x (étape 1 x) est ici la variable booléenne implicite représentant l'état de l'étape CEI 1 dans le module sfc1.

Drapeaux SFC

Pour faciliter le contrôle du déroulement dans le langage SFC, vous pouvez utiliser des drapeaux qui seront générés lors du traitement du projet. À cet effet, vous devez déclarer les variables correspondantes comme étant des variables d'entrée ou de sortie globales ou locales. Exemple : Lorsque dans le SFC une étape dure plus longtemps que ce qu'indiquent ses attributs, un drapeau est utilisé, et vous y avez accès par le biais d'une variable dénommée SFCError (SFCError obtient la valeur TRUE). Les variables suivantes sont possibles pour les drapeaux :

SFCEnableLimit: Cette variable spécifique appartient au type BOOL. Si celle-ci a la valeur TRUE, tous les dépassements de temps au niveau des étapes sont enregistrés dans SFCError. Dans le cas contraire, les dépassements de temps ne sont pas pris en compte.

SFCInit: Si cette variable booléenne est a la valeur TRUE, alors le diagramme fonctionnel en séquence est réinitialisé sur l'étape Init et les autres drapeaux SFC sont réinitialisés (Initialisation). Longtemps que la variable a la valeur TRUE, l'étape Init reste active mais ne sera pas exécutée. Le traitement du module ne reprend normalement qu'à partir du moment où la valeur FALSE est affectée à SFCInit.

SFCReset: Cette variable de type BOOL se comporte comme la variable SFCInit. Elle diffère cependant de celle-ci en ce sens que l'étape Init s'exécute après l'initialisation elle-même. Ceci peut par exemple être mis à profit pour affecter à nouveau la valeur FALSE au SFCReset durant l'étape Init.

SFCQuitError: Aussi longtemps que cette variable booléenne est a la valeur TRUE, l'exécution du diagramme fonctionnel en séquence est suspendue et les dépassements de temps éventuels enregistrés dans la variable SFCError sont réinitialisés. A partir du moment où la valeur FALSE est à nouveau affectée à la variable, tous les temps existants au niveau des étapes actives sont réinitialisés. Une condition pour ce faire est la déclaration du drapeau SFCError qui enregistre les dépassements de temps.

SFCPause. Aussi longtemps que cette variable booléenne est a la valeur TRUE, l'exécution du diagramme fonctionnel en séquence est suspendue.

SFCError: Cette variable booléenne obtient la valeur TRUE lorsqu'un dépassement de temps est intervenu dans un diagramme fonctionnel en séquence. Si, dans le programme, un second dépassement de temps se produit après le premier, il ne sera plus enregistré si la variable SFCError n'a pas auparavant été remise à zéro. La déclaration de SFCError est la condition pour le fonctionnement des autres variables de drapeaux contrôlant le déroulement dans le temps (SFCErrorStep, SFCErrorPOU, SFCQuitError, SFCErrorAnalyzation).

SFCTrans: Cette variable booléenne devient TRUE lorsqu'une transition s'effectue.

SFCErrorStep: Cette variable appartient au type STRING. En cas de dépassement de temps, le nom de l'étape qui a causé le dépassement est mémorisé dans cette variable. Une condition pour ce faire est la déclaration de la variable SFCError qui enregistre les dépassements de temps. Une condition pour ce faire est la déclaration de la variable SFCError qui enregistre les dépassements de temps.

SFCErrorPOU: En cas de dépassement de temps, enregistré par SFCError, le nom du module qui a causé le dépassement est mémorisé dans cette variable de type STRING.

SFCCurrentStep: Cette variable appartient au type STRING. Le nom de l'étape active est stocké dans cette variable indépendamment de la surveillance des temps. Le nom de l'étape de la branche la plus à droite est stocké dans le cas d'une séquence simultanée.

SFCErrorAnalyzationTable: Cette variable de type ARRAY [0..n] OF ExpressionResult donne les informations ci-dessous pour chaque variable d'une expression de transition composée et impliquant le statut FALSE de la transition, ainsi qu'un dépassement de temps de l'étape précédente: nom, adresse, commentaire, valeur actuelle.

Cette fonction est possible pour un maximum de 16 variables, ce qui signifie que la plage d'array est de maximum 0..15.

La structure ExpressionResult ainsi que les modules d'analyse utilisés de manière implicite se trouvent dans la bibliothèque AnalyzationNew.lib. Les modules d'analyse peuvent également être utilisés de manière explicite au sein de modules qui ne sont pas programmés en SFC.

Une condition pour l'analyse de l'expression de transition est l'enregistrement d'un dépassement de temps dans l'étape précédente. C'est pourquoi il faut y implémenter une surveillance et déclarer la variable SFCError (voir plus haut) dans le module.

SFCTip, **SFCTipMode**: Ces variables de type BOOL permettent le fonctionnement par étapes du SFC. Si celui-ci est activé par le biais de SFCTipMode=TRUE, on ne peut passer à l'étape suivante que si SFCTip est positionné sur VRAI. Tant que SFCTipMode est positionné sur FALSE, vous pouvez en outre passer à la suite par le biais des transitions.

Séquence alternative

Dans les réseaux SFC, deux ou plusieurs séquences peuvent être définies comme séquences alternatives. Toute séquence alternative doit commencer et terminer par une transition. Une séquence alternative peut contenir des séquences simultanées et d'autres séquences alternatives. Une séquence alternative commence au niveau d'une ligne horizontale (début de séquences alternatives) et finit au niveau d'une ligne horizontale (fin de séquences alternatives) ou par un saut. Elle peut être munie d'une étiquette de saut.

Si l'étape précédant la ligne de début de séquences alternatives est active, alors la première transition de chaque séquence alternative est évaluée de la gauche vers la droite. La première transition dont la condition de transition a comme valeur TRUE est ouverte, puis les étapes suivantes sont activées (voir étape activée).

Séquence simultanée

Dans les réseaux SFC, deux ou plusieurs séquences peuvent être définies comme séquences simultanées. Toute séquence simultanée doit commencer et terminer par une transition. Une séquence simultanée peut contenir des séquences alternatives et d'autres séquences simultanées. Une séquence simultanée commence au niveau d'une double ligne (début de séquences simultanées) et finit au niveau d'une ligne double horizontale (fin de séquences simultanées) ou par un saut. Elle peut être munie d'une étiquette de saut.

Si l'étape précédant la ligne de début de séquences simultanées est active, et si la condition de transition qui suit cette étape a comme valeur TRUE, alors la première étape de chacune des séquences simultanées est activée (voir étape active). Dès lors, ces séquences sont exécutées parallèlement les unes par rapport aux autres. L'étape qui suit la ligne de fin de séquences simultanées est activée si toutes les étapes précédentes sont actives et si la condition de transition précédant cette étape a la valeur TRUE.

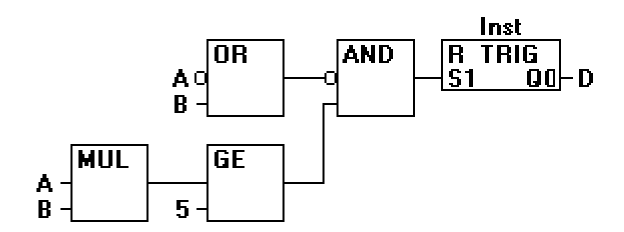
Saut

Un saut constitue une liaison vers l'étape dont le nom est indiqué sous le symbole de saut. Les sauts ont une utilité du fait qu'il n'est pas permis de créer des liaisons vers le haut ni de créer des liaisons qui s'entrecroisent.

2.2.3 Schéma en Blocs Fonctionnels (FBD)

Le Langage en blocs fonctionnels est un langage de programmation graphique. Il fonctionne avec une liste de réseaux, où chaque réseau contient une structure visualisant une expression logique ou arithmétique, un appel de bloc fonctionnel, un saut ou une instruction return.

Exemple pour un réseau en langage FBD



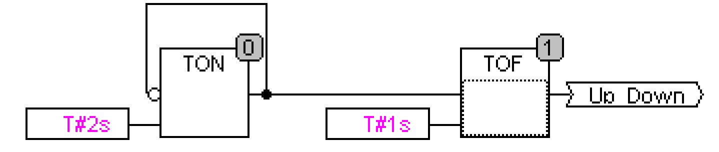
Se reporter pour cela également au chapitre 5.4.2, 'L'éditeur graphique FBD' dans 'Les éditeurs dans CoDeSys'.

2.2.4 Schéma en CFC

L'éditeur graphique CFC ne travaille pas avec des réseaux comme le schéma en plans fonctionnels FBD, mais bien avec des éléments à disposer librement. Ceci permet par exemple des asservissements.

Exemple de réseau en éditeur graphique CFC tel qu'il apparaît dans **CoDeSys** :

Exemple de réseau en éditeur graphique CFC



2.2.5 Langage à Contacts (LD)

Le Langage à contacts est également un langage de programmation graphique, qui se rapproche du principe des schémas électriques.

D'une part, le langage LD est adapté à la construction de dispositifs de commutation logiques et d'autre part, il permet de réaliser des réseaux, comme dans le langage FBD. Par conséquent le langage LD peut parfaitement être utilisé pour commander l'appel d'autres modules. Le langage LD est constitué d'une succession de réseaux. Un réseau LD est délimité sur sa gauche par une barre d'alimentation verticale gauche et sur sa droite par une barre d'alimentation verticale droite. Entre les deux barres d'alimentation se trouve un schéma comprenant des contacts, des bobinages et des éléments de liaison.

La partie gauche de chaque réseau est constituée d'une succession de contacts, qui transmettent de gauche à droite l'état "ON" ou "OFF"; ces états correspondent aux valeurs booléennes TRUE et FALSE. A chaque contact correspond une variable booléenne. Si cette variable a comme valeur TRUE, alors l'état est transmis de la gauche vers la droite le long de l'élément de liaison. Dans le cas contraire, la liaison de droite prend la valeur OFF.

Exemple d'un réseau en langage à contact constitué de contacts et de bobinages:

Contact

En langage LD, la partie gauche de chaque réseau est constitué d'un réseau de *contacts* (un contact est visualisé par deux lignes parallèles : | |), qui transmet de la gauche vers la droite l'état "ON" ou "OFF".

Ces états correspondent aux valeurs booléennes TRUE et FALSE. A chaque contact correspond une variable booléenne. Si cette variable a comme valeur TRUE, alors l'état est transmis de la gauche vers la droite le long de l'élément de liaison. Dans le cas contraire, la liaison de droite prend la valeur "OFF".

Si les contacts sont disposés parallèlement, alors *une* des branches parallèles doit transmettre la valeur "ON" pour que la disposition parallèle transmette la valeur "ON"; si les contacts sont montés en série, alors *tous* les contacts doivent transmettre l'état "ON" pour que le dernier contact transmette l'état "ON". Cela correspond donc respectivement aux montages en parallèle et aux montages en série des circuits électriques.

Un contact peut aussi exister sous forme de négation (contact normalement au travail), ce qui est signalé par la barre oblique à l'intérieur du symbole de contact: |/|. Dans ce cas, la valeur est transmise à l'élément de liaison si la variable a comme valeur FALSE.

Bobinage

En langage LD, on trouve dans la partie droite d'un réseau un nombre indéterminé de "bobinages", visualisés par des parenthèses: **( )**. Ceux-ci peuvent être montés uniquement en parallèle. Un bobinage transmet la valeur de la liaison de gauche à droite, et copie cette valeur dans une variable booléenne associée. La liaison d'entrée peut avoir la valeur ON (correspondant à la variable booléenne TRUE) ou la valeur OFF (correspondant à FALSE).

Les contacts et les bobinages peuvent également exister sous forme de négation (dans l'exemple, le contact SCHALT1 et le bobinage %QX3.0 existent sous forme de négation). Si un bobinage existe sous forme de négation (ce qui est signalé par la barre oblique à l'intérieur du symbole de bobinage: **(/)**), alors celui-ci copie la négation de la valeur dans la variable booléenne associée. Si un contact

existe sous forme de négation, alors celui-ci n'établit la connexion que si la variable booléenne associée a comme valeur FALSE.

Blocs fonctionnels dans le réseau LD

En plus des contacts et des bobinages, vous pouvez aussi introduire des blocs fonctionnels et des programmes. Les blocs fonctionnels et les programmes doivent avoir une entrée et une sortie de réseau de type booléen et peuvent être utilisés aux mêmes endroits que les contacts, c.-à-d. dans la partie gauche du réseau LD.

Bobinage SET/RESET

Il est possible de définir des bobinages SET (verrouillage) ou RESET (déverrouillage). Un bobinage SET (identifié par un "S" à l'intérieur du symbole de bobinage: **(S)**) n'écrase jamais la valeur TRUE dans la variable booléenne associée. En d'autres termes, il suffit que la valeur TRUE ait été affectée une fois à ladite variable pour que cette affectation soit définitive.

Un bobinage RESET (identifié par un "R" à l'intérieur du symbole de bobinage: **(R)**) n'écrase jamais la valeur FALSE dans la variable booléenne associée. En d'autres termes, il suffit que la valeur FALSE ait été affectée une fois à ladite variable pour que cette affectation soit définitive.

Bobinage SET/RESET

Il est possible de définir des bobinages SET (verrouillage) ou RESET (déverrouillage). Un bobinage SET (identifié par un "S" à l'intérieur du symbole de bobinage: **(S)**) n'écrase jamais la valeur TRUE dans la variable booléenne associée. En d'autres termes, il suffit que la valeur TRUE ait été affectée une fois à ladite variable pour que cette affectation soit définitive.

Un bobinage RESET (identifié par un "R" à l'intérieur du symbole de bobinage: **(R)**) n'écrase jamais la valeur FALSE dans la variable booléenne associée. En d'autres termes, il suffit que la valeur FALSE ait été affectée une fois à ladite variable pour que cette affectation soit définitive.

LD sous forme de FBD

En travaillant en langage LD, il se peut facilement que vous souhaitiez utiliser le résultat produit par le montage des contacts pour commander d'autres modules. Dans ce cas, vous pouvez placer à l'aide des bobinages le résultat dans une variable globale, qui sera ensuite utilisée à un autre endroit. Vous pouvez également installer un appel éventuel directement dans votre réseau LD. Pour cela introduisez un module avec entrée EN.

Ces modules sont des opérandes, des fonctions, des programmes ou des blocs fonctionnels tout à fait ordinaires, qui possèdent une entrée supplémentaire portant l'inscription EN. Une entrée EN est toujours de type BOOL et a la signification suivante: le module avec entrée EN est évalué si EN a comme valeur TRUE.

Un module EN est monté en parallèle sur les bobinages, de façon à relier l'entrée EN à l'élément de liaison entre les contacts et les bobinages. Si l'information "ON" est véhiculée le long de cette ligne, alors le module est évalué de façon tout à fait ordinaire.

Un tel module EN peut servir pour réaliser des réseaux comme ceux existant en langage FBD.

Exemple de réseau en langage LD avec module EN:

2.3 Débogage, Fonctionnalités En Ligne

Débogage

Les fonctions de débogage de **CoDeSys** vous aident à détecter les erreurs.

Pour effectuer un débogage, il faut exécuter la commande 'Projet'-"**Options**" et sélectionner ensuite le point **Débogage** dans **Options de compilation**, dans le dialogue qui apparaît.

Point d'arrêt

Un point d'arrêt est un endroit dans le programme au quel l'exécution du programme est interrompue. Cela permet d'observer la valeur des variables à un endroit donné du programme.

Les points d'arrêt peuvent être définis dans n'importe quel éditeur. Les points d'arrêts sont définis au niveau des numéros de ligne dans les éditeurs littéraux, au niveau des numéros de réseau en langage FBD et LD, au niveau du module dans l'éditeur CFC et au niveau des étapes dans le cas des réseaux SFC. On ne peut pas introduire de points d'arrêt dans les instances de blocs fonctionnels.

Pas à pas

Signification du pas à pas:

- en langage IL: exécution du programme jusqu'à la prochaine instruction CAL, LD ou JMP.
- en langage ST: exécuter l'énoncé suivant.
- en langage FBD, LD: exécuter le réseau suivant.
- en SFC: exécuter l'action de la prochaine étape.

Avec une exécution pas à pas, vous pouvez vérifier si la logique de votre programme est correcte.

Cycle par cycle

Si cycle par cycle est sélectionné, l'exécution s'interrompt à la fin de chaque cycle.

Modifier valeurs en ligne

Il est possible d'affecter une valeur donnée à une variable une seule fois en cours d'exécution (ecrire valeur) ou encore d'affecter une valeur donnée à une variable à l'issue de chaque cycle (forcer).

Surveillance

En mode En ligne, toutes les valeurs réelles de l'automate sont lues et représentées pour toutes les variables en cours et visibles à l'écran. Vous trouvez cette représentation dans l'éditeur de déclaration et de programme, et vous pouvez en outre afficher toutes les valeurs réelles de variables par le biais du gestionnaire d'espion et de recettes ou lors d'une visualisation. Si des variables d'instances de blocs fonctionnels sont espionnées, il faut tout d'abord ouvrir l'instance concernée (voir à cet effet : Gestion des objets).

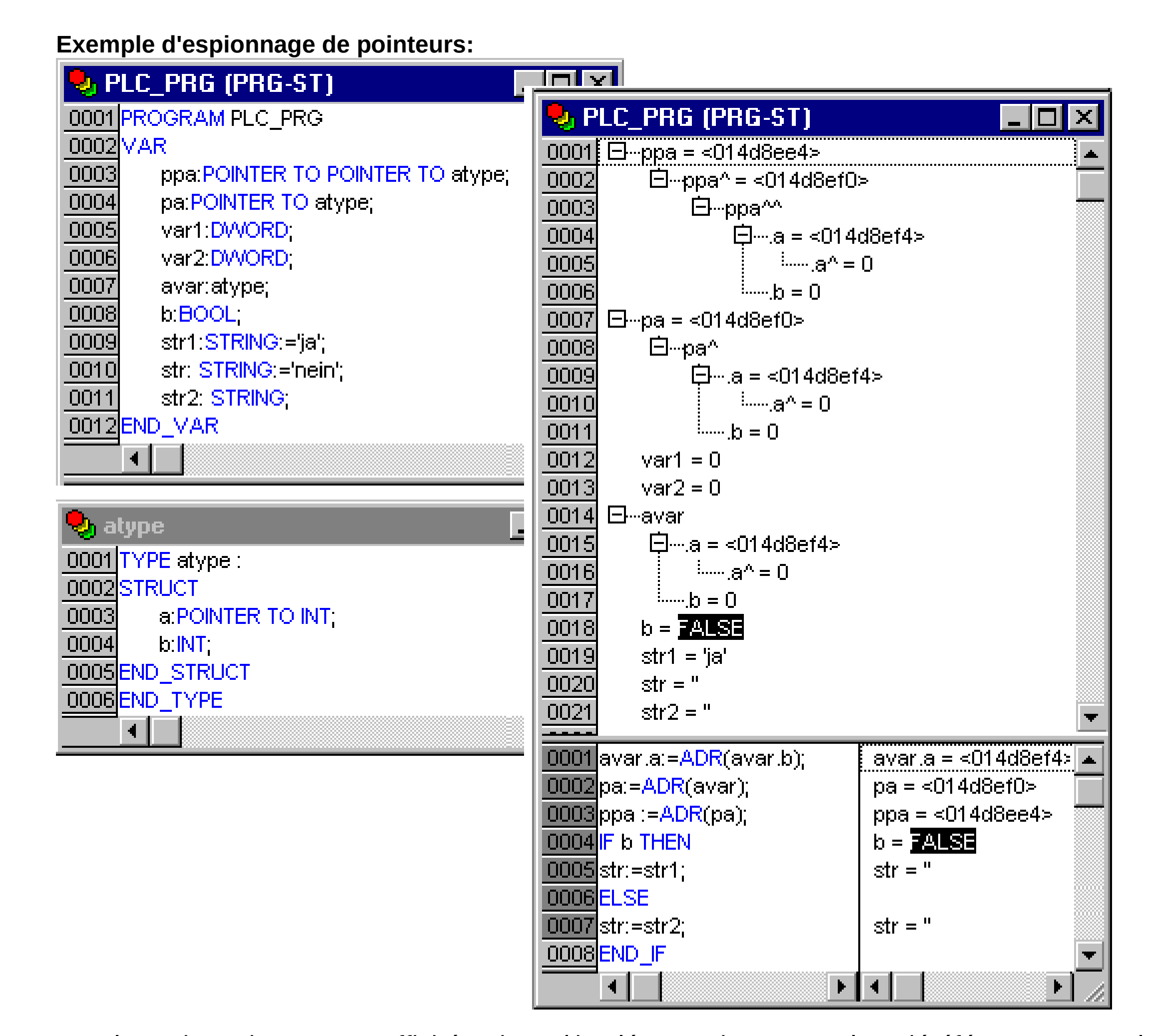
Lors de l'espionnage de variables VAR_IN_OUT, la valeur obtenue par déréférencement est affichée.

Lors de l'espionnage de pointeurs, ceux-ci ainsi que les valeurs obtenues par déréférencement sont affiché(e)s dans la partie déclaration. Dans la partie programme, le pointeur seul est affiché :

```
+ --pointervar = '<'pointervalue'>';
```

Les pointeurs obtenus en déréférençant un pointeur sont également affichés en conséquence. Par un simple clic sur la croix ou double-clic sur la ligne, l'affichage est étendu ou réduit.

Exemple d'espionnage de pointeurs:



La valeur du pointeur est affichée dans l'implémentation. Pour les déréférencements, la valeur déréférencée est affichée.

Espionnage des composants d'un tableau : En plus des composants d'un tableau indexés par une constante, des composants indexés sur une variable sont affichés.

```
anarray[1] = 5
anarray[i] = 1
```

Si l'index se compose d'une expression (p.ex. [i+j] ou [i+1]), les composants ne peuvent être affichés.

Veuillez noter: Lorsque le nombre maximal de variables pouvant être "espionnées" est atteint, le texte "Trop de variables espionnées" est affiché à la place de la valeur actuelle pour chaque variable supplémentaire.

Simulation

Dans le cas de la simulation, le programme d'automate créé est exécuté dans l'ordinateur sur lequel CoDeSys est exécuté et non dans l'automate. Toutes les fonctions En Ligne sont disponibles. Vous avez donc la possibilité de vérifier si la logique de votre programme est correcte sans recourir à un matériel d'automate programmable.

Remarque: Les modules provenant de bibliothèques externes ne fonctionnent pas dans la simulation.

Journal

Lors du mode En Ligne, le journal enregistre chronologiquement les actions de l'utilisateur, les processus internes, les changements d'états et les cas exceptionnels. Il sert à la surveillance et à la traçabilité des erreurs (voir chapitre 'Fonctions En ligne').

2.4 La Norme

La norme CEI 661131-3 est une norme internationale relative aux langages de programmation pour automates programmables.

Les langages de programmation mis en œuvre dans le cadre de **CoDeSys** sont conformes aux exigences de cette norme.

D'après cette norme, un programme est constitué des éléments suivants:

- Structures
- Modules
- Variables globales

L'exécution d'un programme CoDeSys commence par le module particulier PLC_PRG. Le module PLC_PRG peut appeler d'autres modules.

3 Ecrivons un petit programme

3.1 Automatiser un système de feux de signalisation

Introduction

Nous allons à présent commencer à écrire un petit programme d'exemple. Il s'agit de concevoir un mini-système, capable de commander deux feux de signalisation. Les phases rouge/vert de chaque feu alterneront de façon à ce que les phases des deux feux soient toujours en opposition. De plus, afin d'empêcher tout accident, nous prévoirons des phases de transition jaune et jaune/rouge qui marqueront le passage entre les différentes phases. Les phases jaune et jaune/rouge seront d'une durée plus longue que les phases rouge et vert. De plus nous supposons que la commande de feux de signalisation utilise un bus CAN et nous ferons la configuration de l'automate correspondante.

Comment représenter des programmes dépendant du temps en utilisant les outils de langage de la norme CEI61131-3, comment éditer les différents langages de la norme avec l'aide de **CoDeSys**, mais aussi comment les relier entre eux sans problèmes: voilà ce que le programme d'exemple vous fera découvrir. Ce n'est pas tout: vous découvrirez la simulation de **CoDeSys** et vous l'apprécierez à sa juste valeur.

Créer des modules

Le début est toujours facile: commencez par lancer **CoDeSys**, puis sélectionnez 'Fichier' + "Nouveau".

Dans la boîte de dialogue qui apparaît, le nom PLC_PRG a été préaffecté au premier module. Conservez ce nom, sachant que de toutes façons le module doit être du type programme; chaque projet nécessite un programme de ce nom. Dans notre cas, vous sélectionnez l'éditeur FBD graphique (CFC) comme langage de programmation pour ce module.

Créez à présent trois objets supplémentaires au moyen de la commande 'Projet' + `Insérer objet' de la barre de menus ou du menu contextuel (cliquer avec le bouton droit de la souris dans l'Organisateur d'objets): un programme en diagramme fonctionnel en séquence (SFC) nommé SEQUENCE, un bloc fonctionnel en langage FBD nommé TRAFFICSIGNAL et un module nommé WAIT, également du type bloc fonctionnel, que nous souhaitons programmer en langage Liste d'Instructions (IL).

Quel est le rôle de TRAFFICSIGNAL?

Dans le module TRAFFICSIGNAL nous allons affecter les différentes phases de feux aux lampes des feux, c.-à-d. que nous veillerons à ce que la lampe rouge s'allume pendant les phases rouge et jaune/rouge, à ce que la lampe jaune s'allume pendant les phases jaune et jaune/rouge, etc.

Quel est le rôle de WAIT?

Dans WAIT nous allons programmer un temporisateur simple, qui reçoit comme entrée la durée de la phase en millisecondes et qui fournit comme sortie la valeur TRUE dès que le temps est écoulé.

Quel est le rôle de SEQUENCE?

SEQUENCE reliera tous les éléments de façon à ce que la bonne lampe s'allume au moment voulu et pendant le temps voulu.

Quel est le rôle de PLC_PRG?

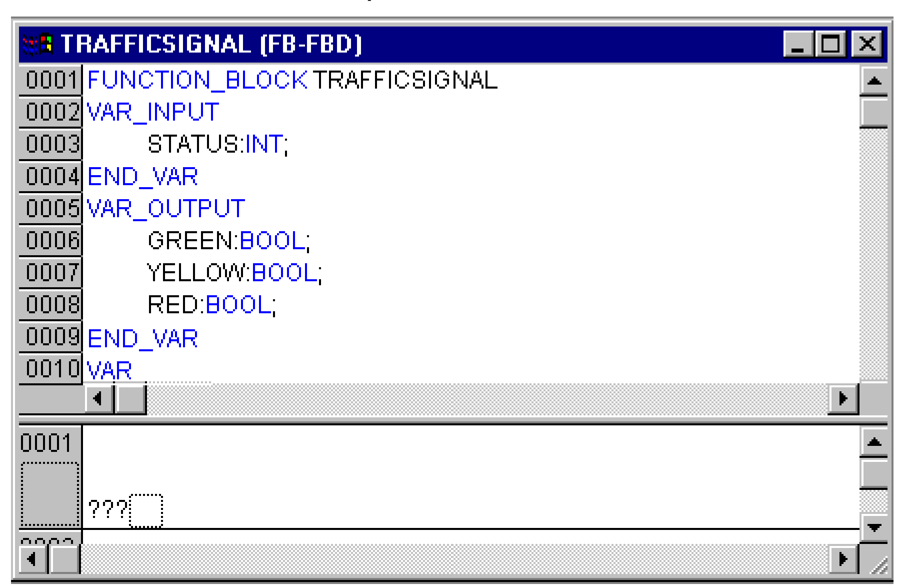
PLC_PRG reliera le signal de départ entrant avec la séquence de déroulement des phases des feux de signalisation et fournira les "instructions d'allumage" des différentes lampes des deux feux de signalisation sous forme de sorties.

Déclaration de "TRAFFICSIGNAL"

Consacrons-nous d'abord au module TRAFFICSIGNAL. Dans l'éditeur de déclaration vous déclarez comme variable d'entrée une variable nommée STATUS, de type INT (entre les mots-clés VAR_INPUT et END_VAR). STATUS pourra adopter quatre états différents, correspondant respectivement aux phases de feux vert, jaune, jaune-rouge et rouge.

Notre feu possède par conséquent trois sorties, à savoir RED, YELLOW et GREEN. (Attention : des trémas ne sont pas permis dans les noms de variables!) Déclarez ces trois variables et la partie de déclaration de notre bloc fonctionnel TRAFFICSIGNAL ressemblera alors à ceci:

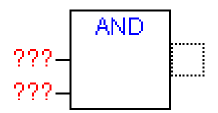
Bloc fonctionnel TRAFFICSIGNAL, partie de déclaration



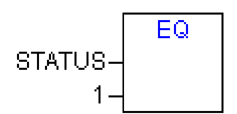
Corps de "TRAFFICSIGNAL"

A présent, il s'agit de calculer les valeurs des variables de sortie en fonction de l'entrée STATUS pour le module. Accédez à la partie corps du module. Cliquez dans la zone située à gauche du premier réseau (la zone grise portant le numéro 0001). Vous avez sélectionné le premier réseau. Sélectionnez maintenant l'option 'Insérer' + `Module'

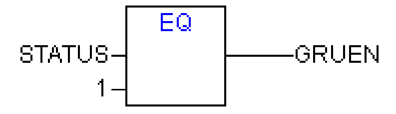
Une boîte avec l'opérateur AND et deux entrées est insérée dans le premier réseau:



Cliquez sur le texte AND avec le pointeur de la souris et remplacez ce texte par EQ. Sélectionnez les trois points d'interrogation de l'entrée supérieure et entrez la variable STATUS. Ensuite, sélectionnez les trois points d'interrogation du dessous et remplacez les par 1. Vous obtenez le réseau suivant:



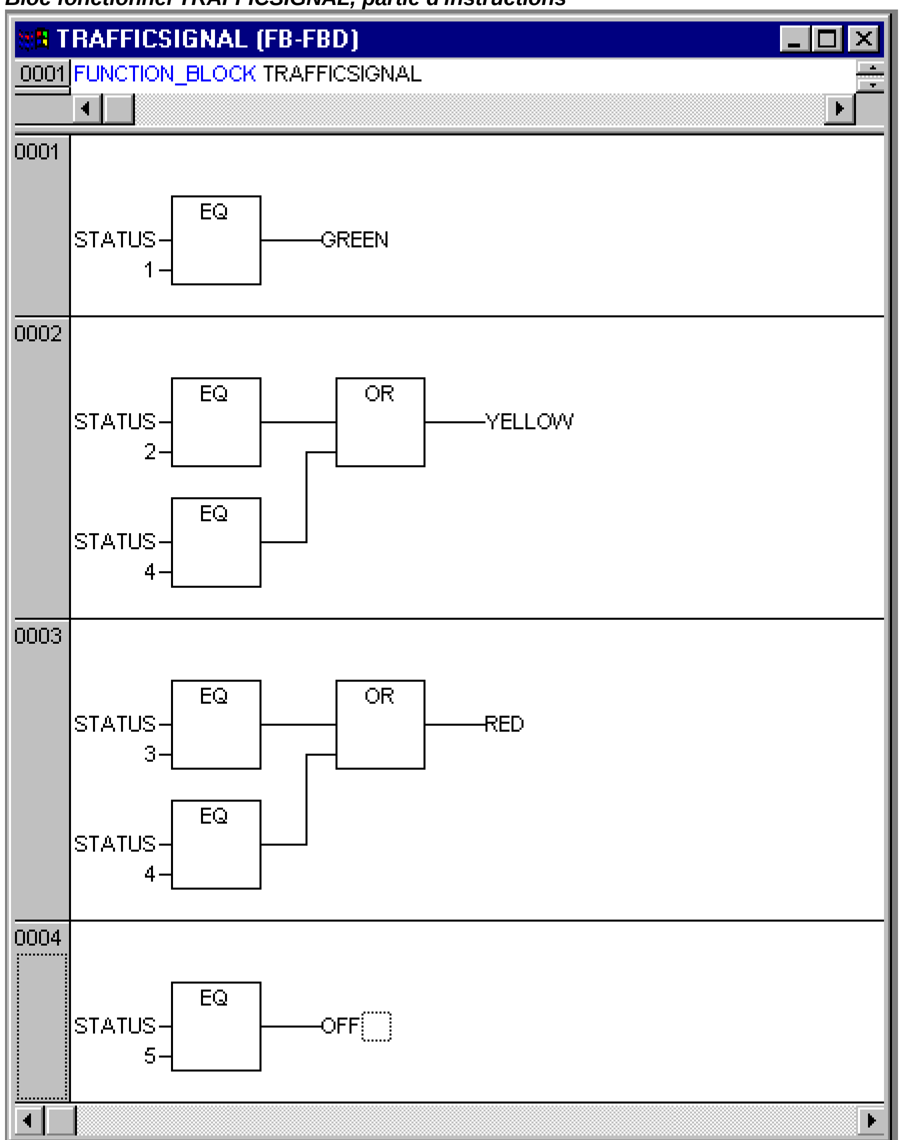
Cliquez à présent à un endroit quelconque derrière la boîte EQ. La sortie de l'opération EQ est maintenant sélectionnée. Sélectionnez 'Insérer' + 'Affectation'. Remplacez les trois points d'interrogation ??? par GREEN. Vous avez créé un réseau de cette forme:



STATUS est comparé à 1 et le résultat est affecté à GREEN. Ce réseau commute sur GREEN lorsque STATUS a la valeur 1.

Pour les autres couleurs de feux, nous avons besoin de deux réseaux supplémentaires. Créez le premier avec la commande 'Insérer' + "Réseau (derrière)" et créez un module EQ comme décrit ci-dessus. Lorsque vous avez sélectionné la sortie, choisissez la commande 'Insérez' 'Module' et remplacez le "AND" par un "OR" dans celle-ci. Sélectionnez à nouveau la sortie du module OR et affectez lui YELLOW par le biais de la commande **'Insérer' 'Affectation'**. Sélectionnez maintenant la seconde entrée du OR en cliquant sur la barre horizontale à côté des trois points d'interrogation, si bien que celle-ci est maintenant marquée d'un rectangle pointillé, puis insérez un autre module EQ comme décrit ci-dessus au moyen de la commande **'Insérer' 'Module'.** En fin de compte, le réseau devrait se présenter comme indiqué à la figure 3.2.

Bloc fonctionnel TRAFFICSIGNAL, partie d'instructions



Pour insérer un nouvel opérateur devant un opérateur existant, vous devez sélectionner l'entrée à laquelle vous voulez ajouter l'opérateur, puis sélectionner l'endroit précis où l'entrée aboutit à la boîte.

Ensuite, exécutez la commande 'Insérer' + `Module'. Pour le reste, vous pouvez procéder à l'élaboration des autres réseaux de la même façon que pour celui-ci.

Notre premier module est à présent terminé. TRAFFICSIGNAL commande la couleur de feu appropriée, en fonction de la valeur qui a été entrée dans STATUS.

Intégrer la bibliothèque standard.lib

Pour le temporisateur du module WAIT, nous avons besoin d'un module issu de la bibliothèque standard. Ouvrez le gestionnaire de bibliothèques au moyen de 'Fenêtre' + `Gestion des

bibliothèques'. Sélectionnez 'Insérer' + ‘Autre bibliothèque’. La boîte de dialogue d'ouverture des fichiers apparaît. Dans la liste des bibliothèques, sélectionnez standard.lib.

Déclaration de "WAIT"

Intéressons-nous à présent au module WAIT. Ce module doit jouer le rôle d'un temporisateur, au moyen duquel nous pouvons spécifier la durée de chaque phase des feux. Notre module reçoit comme entrée une variable nommée TEMPS du type TIME et fournit comme sortie un élément booléen que nous appellerons OK; cet élément aura pour valeur TRUE lorsque le temps prescrit sera écoulé. Nous attribuons par défaut la valeur FALSE à cet élément, en insérant " := FALSE " à la fin de la déclaration (mais avant le point-virgule).

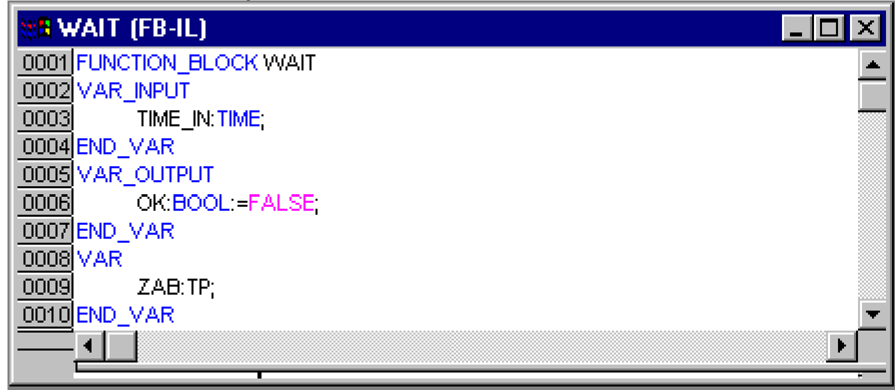
Pour nos besoins, nous utilisons un module TP, un générateur d'impulsions. Celui-ci possède deux entrées (IN, PT) et deux sorties (Q, ET). Voici ce qu'effectue TP:

Aussi longtemps que IN a comme valeur FALSE, ET a comme valeur 0 et Q a comme valeur FALSE. Dès que IN fournit la valeur TRUE, le temps est incrémenté dans la variable de sortie ET, avec des millisecondes comme unité d'incrémentation. Lorsque ET atteint la valeur PT, l'incrémentation cesse. Entre-temps, Q fournit comme valeur TRUE aussi longtemps que ET demeure inférieur à PT. Dès que la valeur PT est atteinte, Q fournit à nouveau comme valeur FALSE.

Pour pouvoir utiliser le module TP dans le module WAIT, nous devons créer une instance locale de TP. Pour cela, nous déclarons une variable locale ZAB (pour le temps écoulé) de type TP (entre les mots-clés VAR, END_VAR).

La partie de déclaration de WAIT ressemble par conséquent à ceci:

Bloc fonctionnel WAIT, partie de déclaration



Corps de "WAIT"

Pour réaliser le temporisateur voulu, il faut que le corps du module soit programmé comme suit:

Pour commencer, un contrôle est effectué pour savoir si Q est déjà TRUE (on veut savoir si le comptage a déjà commencé). Si c'est le cas, nous ne modifions en rien l'affectation de ZAB, et appelons le bloc fonctionnel ZAB sans fournir d'entrée (pour vérifier si le temps est déjà écoulé).

Dans le cas contraire, nous affectons la valeur FALSE à la variable IN dans ZAB, ce qui implique simultanément l'affectation de 0 à ET et de FALSE à Q. Toutes les variables se trouvent ainsi à l'état initial voulu. A présent, nous transférons la durée nécessaire depuis la variable TIME vers la variable PT, et appelons ZAB avec IN:=TRUE. Au niveau du bloc fonctionnel ZAB, la variable ET est incrémentée jusqu'à ce qu'elle atteigne la valeur TIME, puis la valeur FALSE est affectée à Q.

Bloc fonctionnel WAIT, partie d'instructions

WAIT (FB-IL)

```
FUNCTION_BLOCK WAIT
      LD        ZAB.Q
      JMPC      mark

      CAL       ZAB(IN:=FALSE)
      LD        TIME_IN
      ST        ZAB.PT
      CAL       ZAB(IN:=TRUE)
      JMP       end

mark:
      CAL       ZAB
end:
      LDN       ZAB.Q
      ST        OK
      RET
```

La valeur inverse de Q est mémorisée dans OK à chaque fois que WAIT a été parcouru une fois. Dès que Q a comme valeur FALSE, OK fournit donc comme valeur TRUE.

Nous avons ainsi terminé le temporisateur. A présent, il s'agit de réunir nos deux blocs fonctionnels WAIT et TRAFFICSIGNAL dans le programme principal SEQUENCE.

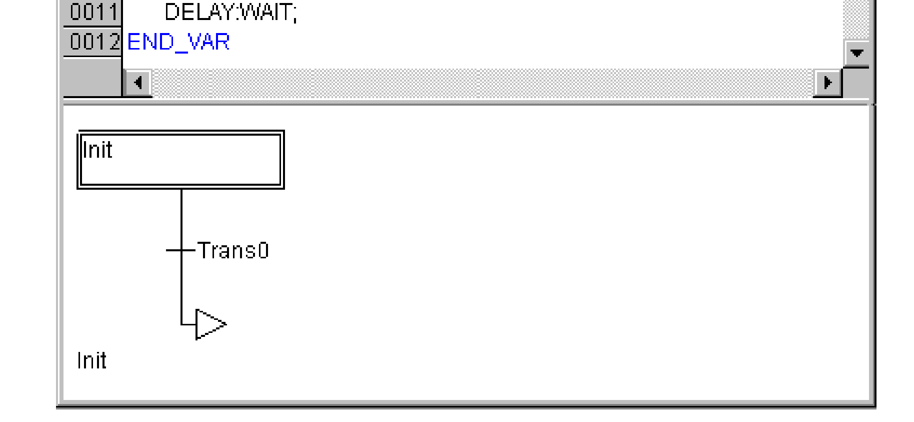
"SEQUENCE" - premier niveau de développement

Pour commencer, nous déclarons les variables dont nous avons besoin. C'est-à-dire une variable d'entrée START du type BOOL, deux variables de sorties TRAFFICSIGNAL1 et TRAFFICSIGNAL2 du type INT et une de type WAIT (DELAY pour la temporisation). Le programme SEQUENCE ressemble maintenant à ceci:

Programme SEQUENCE, premier niveau de développement, partie de déclaration

SEQUENCE (PRG-SFC)

```
PROGRAM SEQUENCE
VAR_INPUT
    START:BOOL;
END_VAR
VAR_OUTPUT
    TRAFFICSIGNAL1:INT;
    TRAFFICSIGNAL2:INT;
END_VAR
VAR
    COUNTER:INT;
    DELAY:WAIT;
END_VAR
```



Créer un diagramme SFC

Les diagrammes initiaux d'un module en langage SFC sont toujours constitués d'une action "Init", suivie d'une transition "Trans0" et d'un saut pour retourner à Init. Il convient de développer un peu cela.

Définissons d'abord la structure du diagramme, avant de programmer les actions et les transitions une par une. Pour commencer, nous avons besoin d'une étape pour chaque phase des feux. Insérez cette étape en marquant Trans0 puis en sélectionnant 'Insérer' + "'Etape-Transition (derrière)". Répétez ce procédé trois fois.

En cliquant directement sur le nom d'une transition ou d'une étape, vous marquez celle-ci et vous pouvez la modifier. Attribuez le nom "START" à la première transition après Init et le nom "DELAY.OK" à toutes les autres transitions.

La première transition s'effectue lorsque START prend la valeur TRUE et les autres s'effectuent uniquement lorsque la variable OK de DELAY est TRUE, c.-à-d. lorsque le temps entré s'est écoulé.

Les étapes reçoivent les noms Switch1, Green2, Switch2, Green1 (de haut en bas) tandis que Init conserve évidemment son nom. "Switch" désigne à chaque fois une phase jaune, Green1 indique que SIGNAL1 est vert, Green2 indique que SIGNAL2 est vert. Pour finir, modifiez l'adresse de saut de retour de Init vers Switch1. Si vous avez procédé sans erreur, le diagramme devrait maintenant ressembler à ceci:

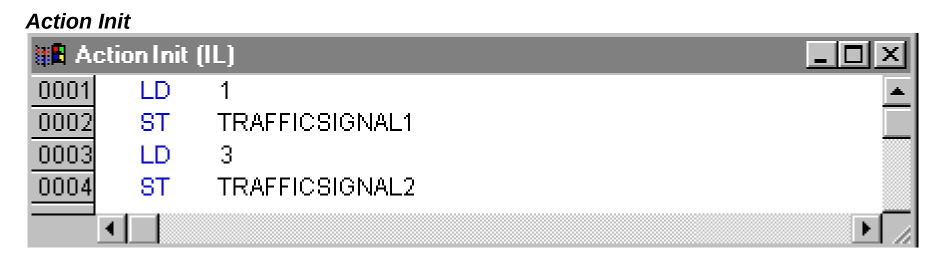
Programme SEQUENCE, premier niveau de développement, partie d'instructions

Nous devons à présent réaliser la programmation des étapes une à une. En double-cliquant sur la zone d'une étape, vous ouvrez une boîte de dialogue pour ouvrir une nouvelle action. Dans le cas présent, nous utiliserons à chaque fois le langage IL (Liste d'Instructions) actions et conditions de transition

Actions et conditions de transition

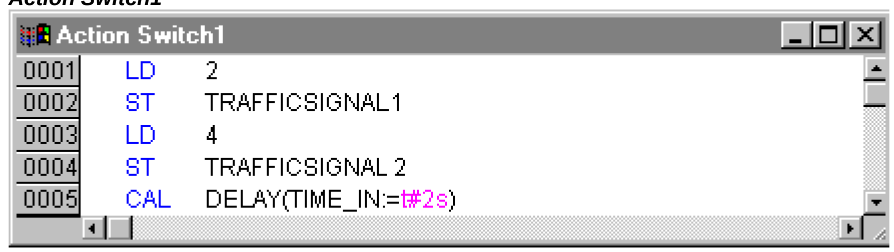
Au niveau de l'action relative à l'étape **Init** les variables sont initialisées, STATUS de TRAFFICSIGNAL1 doit être 1 (vert), STATUS de TRAFFICSIGNAL2 doit être 3 (rouge). L'action Init ressemble alors à ceci:

Action Init



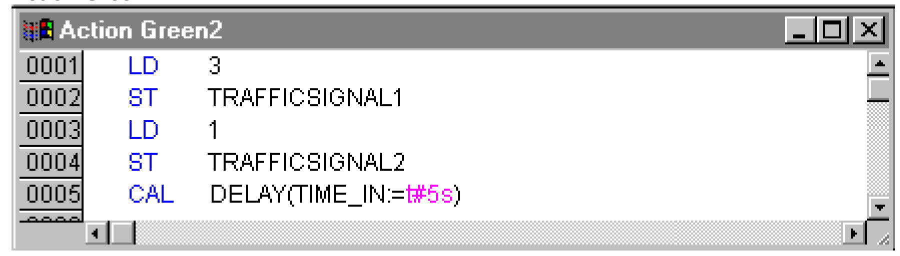
Au niveau de **Switch1** STATUS de TRAFFICSIGNAL1 change d'état, à savoir 2 (jaune), STATUS de TRAFFICSIGNAL2 change également d'état, à savoir 4 (jaune-rouge). En plus, une temporisation de 2000 millisecondes est définie. L'action ressemble à présent à ceci:

Action Switch1



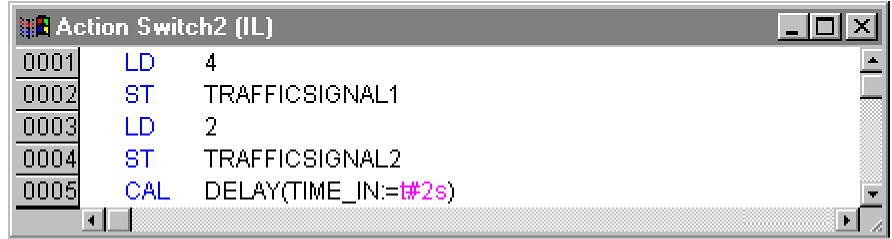
Au niveau de **Green2** TRAFFICSIGNAL1 est rouge (STATUS:=3), TRAFFICSIGNAL2 est vert (STATUS:=1) et la durée de temporisation est de 5000 millisecondes.

Action Green2



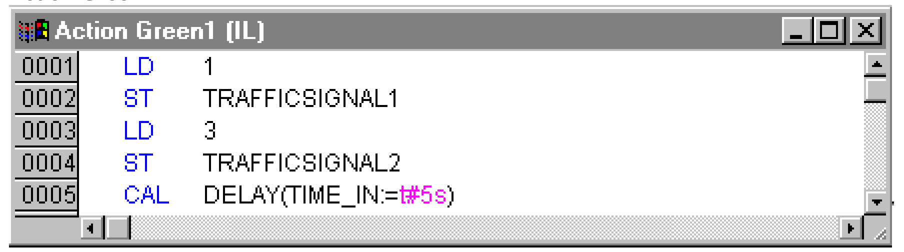
Au niveau de **Switch2** STATUS de TRAFFICSIGNAL1 change d'état, à savoir 4 (jaune-rouge), STATUS de TRAFFICSIGNAL2 change également d'état, à savoir 2 (jaune). Une durée de temporisation de 2000 millisecondes est maintenant définie.

Action Switch2



Au niveau de **Green1** SIGNAL1 est vert (STATUS:=1), SIGNAL2 est rouge (STATUS:=3) et la durée de temporisation est de 5000 millisecondes.

Action Green1



Le premier niveau de développement de notre programme est à présent terminé. Vous pouvez maintenant le compiler et le tester dans la simulation.

Pour cela, le projet doit être compilé par le biais de la commande 'Projet' 'Compiler'. Vous devriez avoir l'indication "0 erreurs, 0 avertissements" dans la fenêtre des messages située sous la fenêtre de travail. Exécutez maintenant la commande 'En Ligne' 'Accéder au système', pour accéder au mode de simulation (l'option 'En Ligne' 'Simulation' devrait déjà être activée). Démarrez le programme avec 'En Ligne' 'Démarrer'. Ouvrez le module SEQUENCE, en double-cliquant sur « SEQUENCE » dans l'Organisateur d'objets. Le programme est maintenant démarré, mais pour débuter le fonctionnement des feux, il faut encore que la variable START soit affectée de la valeur TRUE (vrai). Elle recevra cette valeur plus tard par le biais de PLC_PRG, mais pour l'instant, nous devons encore la lui attribuer directement dans le module. Double-cliquez à cet effet sur la ligne dans laquelle START est défini (START = FALSE) dans la partie déclaration de SEQUENCE. Suite à quoi l'option "<:=TRUE>" apparaît en couleur turquoise derrière la variable. Sélectionnez la commande 'En Ligne' 'Ecrire valeurs des variables', pour régler la variable sur cette valeur. START apparaît alors en bleu sur le diagramme de déroulement, et vous pouvez identifier le traitement de chaque étape individuellement grâce au marquage en bleu de l'étape en cours.

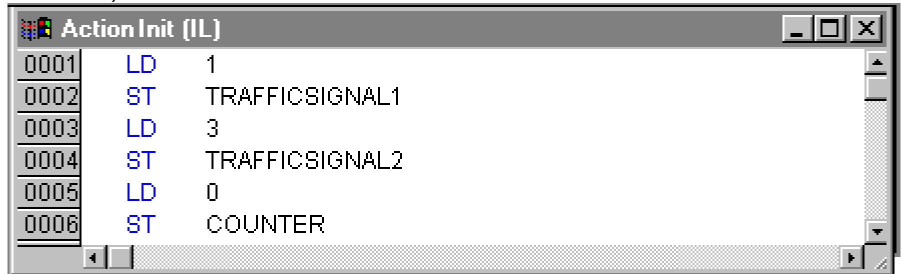
Le test intermédiaire est alors terminé. Effectuez ensuite la commande 'En Ligne' 'Quitter le système' afin de pouvoir quitter le mode simulation et continuer à programmer.

"SEQUENCE" - deuxième niveau de développement

Afin que notre diagramme comporte au moins une séquence alternative et pour pouvoir arrêter notre système la nuit, nous installons maintenant dans notre programme un compteur qui arrête le système après que les feux aient parcouru un nombre donné de cycles.

Nous avons d'abord besoin d'une nouvelle variable COUNTER de type INT. Déclarez cette variable en procédant comme vu précédemment dans la partie de déclaration de SEQUENCE et initialisez-la à la valeur 0, dans Init.

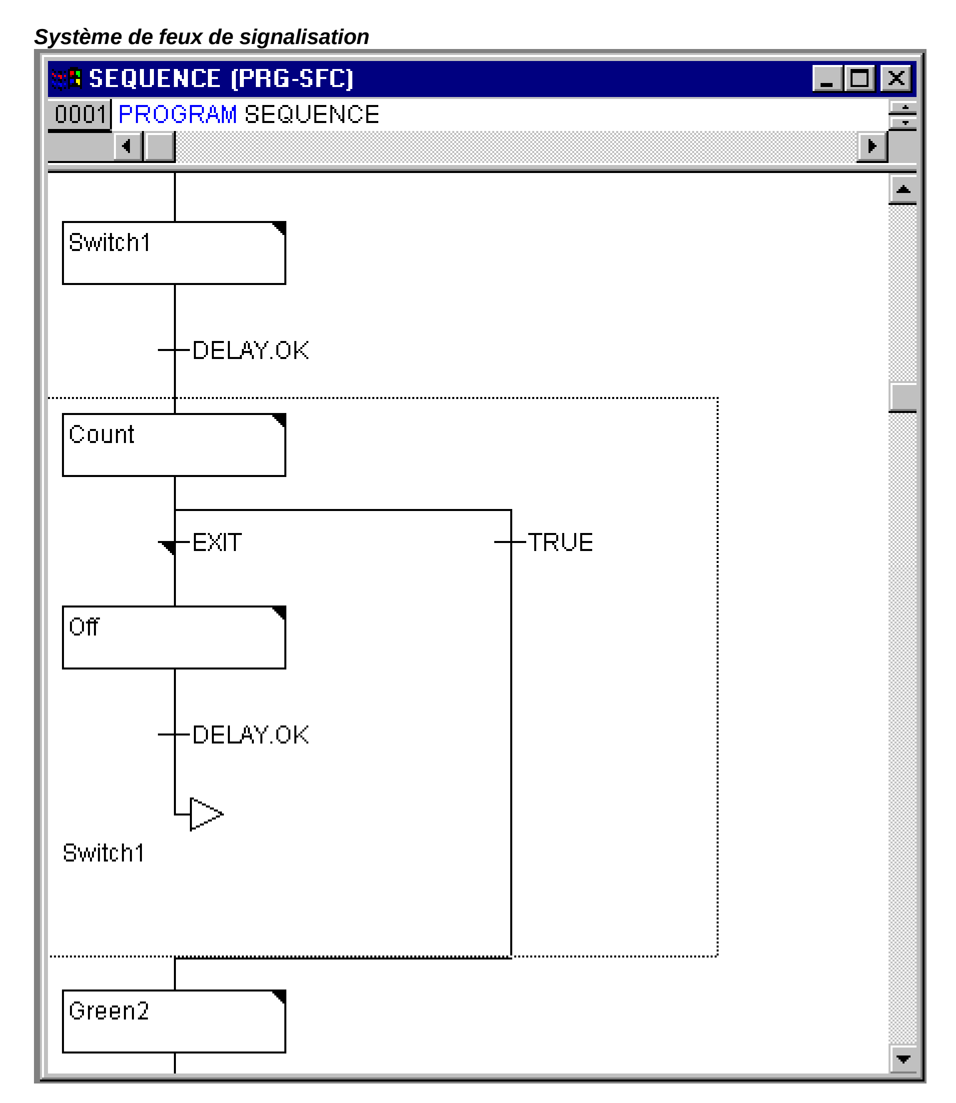
Action Init, deuxième version



Marquez à présent la transition vers Switch1 et insérez à sa suite une étape et une transition. Marquez la transition qui vient d'être créée et insérez sur sa gauche une séquence alternative. Insérez une étape et une transition après la transition de gauche. Insérez un saut vers Switch1 après la transition ainsi créée.

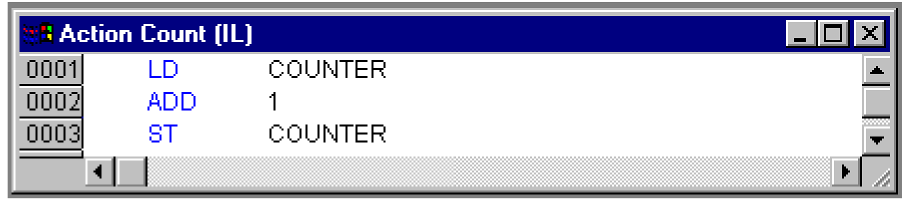
Nommez les composants nouvellement créés comme suit: en ce qui concerne les deux étapes nouvellement créées, l'étape supérieure doit recevoir le nom "Count" et l'étape inférieure le nom "Off". Les transitions sont nommées (de haut en bas et de gauche à droite) EXIT, TRUE et DELAY.OK. La partie nouvellement créée devrait donc ressembler à la partie encadrée en noir et représentée ci-dessous:

Système de feux de signalisation



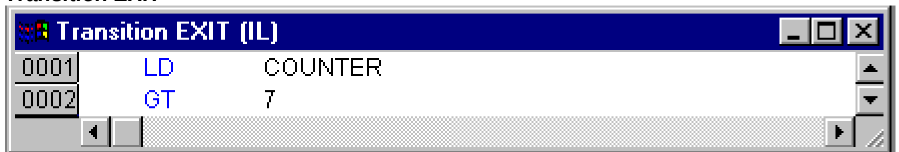
Il faut donc implémenter deux nouvelles actions et une nouvelle condition de transition. Au niveau de l'étape Count, le COUNTER est incrémenté d'une unité; il s'agit de la seule intervention à ce niveau:

Action Count



La transition EXIT vérifie si le compteur (COUNTER) est supérieur à un nombre donné, disons par exemple 7:

Transition EXIT



Au niveau de l'action Off, les variables d'état STATUS des deux feux prennent la valeur 5 (OFF) (toute autre valeur différente de 1, 2, 3 ou 4 peut être choisie pour définir cet état), COUNTER est remis à zéro et une durée de temporisation de 10 secondes est définie:

Action Off

```
Action Off (IL)
0001  LD   5
0002  ST   TRAFFICSIGNAL1
0003  LD   5
0004  ST   TRAFFICSIGNAL2
0005  LD   0
0006  ST   COUNTER
0007  CAL  DELAY(TIME_IN:=t#10s)
```

Le Résultat

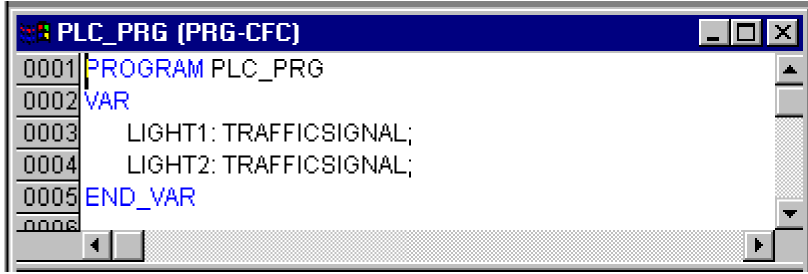
Dans notre petite agglomération, où sont installés les feux, la nuit survient après que les feux aient parcouru sept cycles, ensuite le feu s'éteint pendant dix secondes, puis le jour se lève, le système de feux se met de nouveau en marche et tout ce cycle complet est répété. Si vous le souhaitez, vous pouvez maintenant le tester en mode de simulation comme décrit ci-dessus, avant de créer le module PLC_PRG.

PLC_PRG

Nous avons défini et mis en corrélation l'évolution des phases des deux feux de signalisation dans le module SEQUENCE. Cependant, du fait que nous assimilions le système de feux de signalisation à un module CAN et que nous souhaitions créer la configuration d'automate en utilisant un bus CAN, nous devons encore prévoir des variables d'entrée et de sortie appropriées au sein du module PLC_PRG. Nous souhaitons mettre en marche le système de feux de signalisation au moyen d'un interrupteur ON et transmettre l'„instruction de signal" adéquat pour chaque étape du module SEQUENCE aux six différentes lampes (lampes rouge, verte et jaune pour chaque feu de signalisation). Nous déclarons à présent – avant de créer le programme dans l'éditeur – des variables de type booléen appropriées aux six sorties et à l'unique entrée. Dans le même temps, nous affectons ces variables aux adresses CEI appropriées.

Dans l'éditeur de déclaration de PLC_PRG, nous déclarons dans un premier temps les variables LIGHT1 et LIGHT2 de type TRAFFICSIGNAL.

Déclaration de SIGNAL1 et SIGNAL2

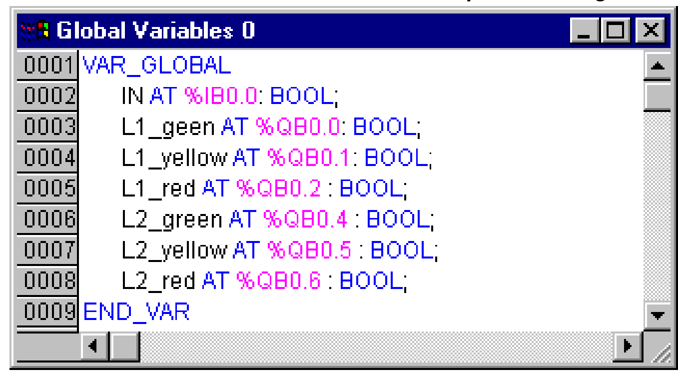


Ces variables transmettent aux six sorties exigées ci-dessus les valeurs booléennes pour chacune des six lampes, à chaque étape du module SEQUENCE. Cependant, nous ne déclarons pas les six variables de sortie prévues à cet effet à l'intérieur du module PLC_PRG, mais bien dans Ressources, au niveau des Variables Globales. La même remarque s'applique à la variable d'entrée ON de type booléen, qui permet d'affecter la valeur TRUE à la variable START dans le module SEQUENCE. Une adresse CEI est attribuée aussi à la variable ON.

Sélectionnez à présent l'onglet Ressources et ouvrez la liste Variables Globales.

Effectuez la déclaration comme ceci:

Déclaration des variables d'entrée / de sortie pour la configuration CAN



Le nom de la variable (par exemple ON) est suivi de AT et de l'adresse CEI, qui commence par un signe „ pour cent ". I désigne une entrée, Q désigne une sortie, B (utilisé dans cet exemple) signifie "octet" et 0.0 (0.1, 0.2, etc.) permet d'accéder à chacun des bits du module. Nous allons créer la configuration de l'automate requise, mais avant cela nous allons terminer le module PLC_PRG.

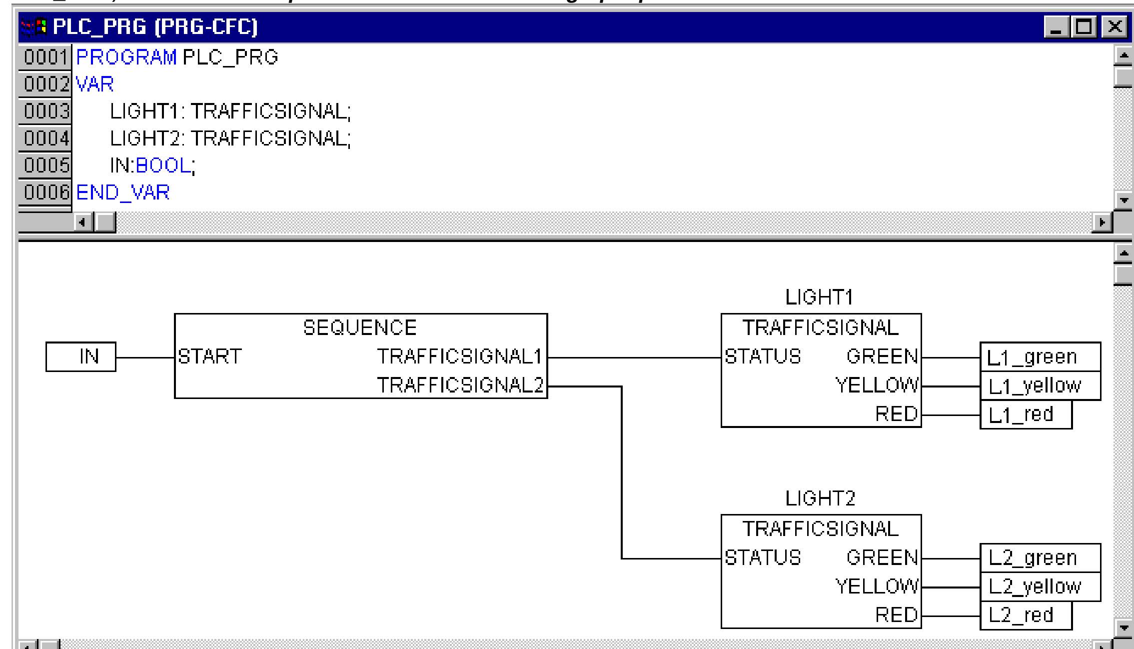
Pour ce faire, nous accédons à la fenêtre d'édition. Nous avons sélectionné l'éditeur graphique FBD, c'est pourquoi nous accédons à la barre d'icônes FBD, qui est située sous la barre de menu et qui comprend les éléments pouvant être utilisés comme composants .

Cliquez avec la touche droite de la souris dans la fenêtre d'édition et sélectionnez l'élément **Module**. Cliquez sur le teste AND et remplacez-le par le texte "SEQUENCE". Vous accédez alors au module SEQUENCE qui est visualisé avec les variables d'entrée et de sortie déjà définies. Ajoutez deux éléments de type module supplémentaires, auxquels vous donnerez le nom de LIGHT1. SIGNAL est un bloc fonctionnel, c'est pourquoi trois points d'interrogation rouges sont placés au-dessus du module. Remplacez ces points d'interrogation par le nom des variables LIGHT1 et LIGHT2 déclarées localement ci-dessus. Définissez à présent un élément de type **Entrée**, que vous désignerez par ON et six éléments de type **Sortie**, auxquels vous donnerez les noms de variable S1_green, S1_yellow, S1_red, S2_green, S2_yellow, S2_red, comme dans la figure ci-après.

Tous les éléments du programme sont à présent placés et vous pouvez relier vos entrées et vos sorties en cliquant avec la souris sur la courte ligne située à l'entrée ou à la sortie d'un élément et en déplaçant ensuite la souris jusqu'à l'entrée ou la sortie de l'élément voulu, tout en maintenant la touche de la souris enfoncée. Votre ligne est tracée.

En définitive, votre programme doit se présenter comme ceci:

PLC_PRG, déclaration et représentation dans l'éditeur graphique FBD



Simulation de feux

Testez à présent votre programme. Pour cela, il vous faut le compiler compiler) et le charger ('En Ligne' + Accéder au système') et puis 'En Ligne' + 'Lancer'. Si vous exécutez maintenant 'En Ligne' + 'Démarrer', vous pouvez suivre la succession chronologique des étapes individuelles de votre programme principal. La fenêtre du module PLC_PRG s'est à présent transformée en fenêtre d'espion. Lorsque vous double-cliquez sur le signe plus dans l'éditeur de déclaration, la visualisation des variables apparaît, et vous pouvez observer les valeurs prises par chacune des variables.

3.2 La visualisation du système de feux de signalisation

La visualisation incluse dans **CoDeSys** permet d'animer facilement et simplement les variables du projet. Dans la section présente, nous allons dessiner deux feux de signalisation et un interrupteur de mise en marche (ON) pour notre système de feux, qui sont censés représenter les séquences de commutation.

Elaboration d'une nouvelle visualisation

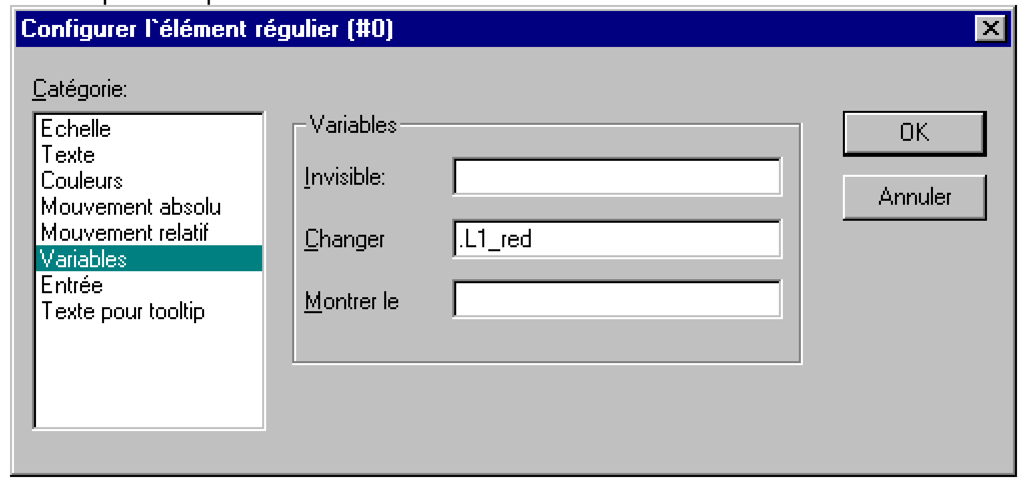
Pour élaborer une visualisation, il faut tout d'abord sélectionner la zone **Visualisation** dans l'Organisateur d'objets. Pour cela, cliquez d'abord sur l'onglet nommé **Visualisation** et portant le symbole au niveau du cadre inférieur de la fenêtre située sur le côté gauche, dans laquelle est écrit **Modules**. Exécutez maintenant l'ordre 'Projet' + `Insérer objet' et une boîte de dialogue s'ouvre.

Entrez un nom de votre choix à cet endroit. Confirmez par **OK** le choix fait dans cette boîte de dialogue et une fenêtre s'ouvre, dans laquelle vous pouvez élaborer votre nouvelle visualisation.

Insérer un élément dans la visualisation

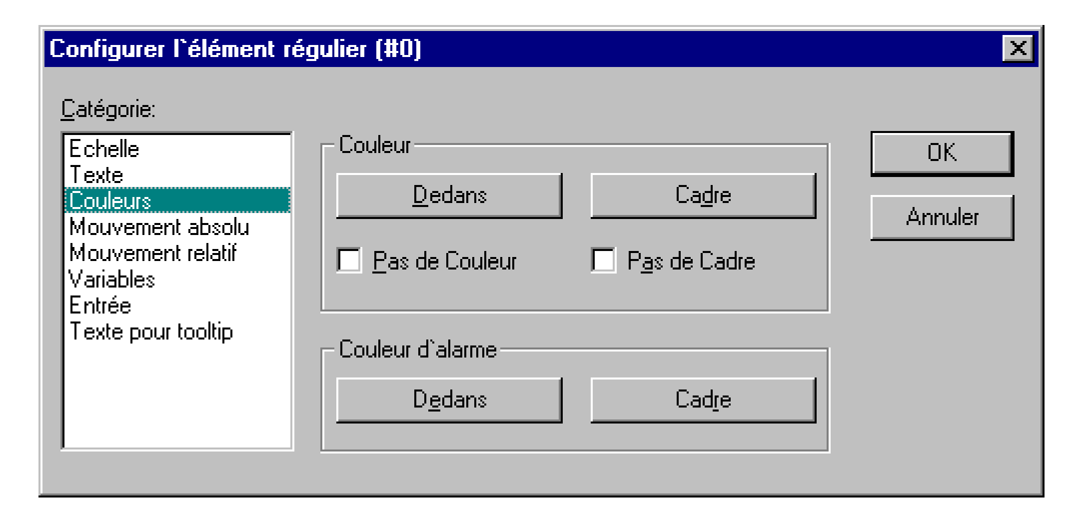
Pour la visualisation qui concerne nos feux, nous vous conseillons de procéder comme suit:

- Activez la commande 'Insérer' + 'Ellipse' et essayez de dessiner un cercle qui ne soit pas trop grand (□ 2 cm). Pour ce faire, cliquez dans le champ d'édition et tracez le cercle en déplaçant la souris tout en maintenant la touche gauche enfoncée.
- Effectuez maintenant un double-clic sur le cercle. La boîte de dialogue destinée à éditer des éléments de visualisation s'ouvre alors.
- Choisissez la catégorie Variables et entrez dans le champ Changer le texte "L1_red" ou ".L1_red". Cela signifie que la variable globale A1_red effectue le changement de couleur si elle prend la valeur TRUE. Le point avant le nom de la variable indique qu'il s'agit d'une variable globale mais n'est cependant pas absolument nécessaire.



- Ensuite, vous sélectionnez la catégorie Couleurs et vous cliquez sur le bouton Dedans dans la zone Couleur. Sélectionnez une couleur neutre, par exemple noir.
- Cliquez à présent sur le bouton **Dedans** dans la zone **Couleur d'alarme** et sélectionnez le ton rouge qui correspond le mieux à un feu rouge.





Le cercle ainsi constitué sera noir à l'état normal et passera au rouge lorsque la variable RED de SIGNAL1 aura comme valeur TRUE. Nous avons donc élaboré dès à présent la première lampe du premier feu!

Les autres lampes des feux

Sélectionnez maintenant les commandes 'Editer' + "Copier" (<Ctrl>+<C>), puis deux fois 'Editer' + "Coller" (<Ctrl>+<V>). Ainsi, vous obtenez deux autres cercles de même dimension, qui sont situés au-dessus du premier. Vous pouvez déplacer les cercles en cliquant sur le cercle voulu et en déplaçant la souris jusqu'à la position voulue, tout en maintenant la touche gauche de la souris enfoncée. Dans le cas présent, la position voulue correspondrait à une rangée verticale dans la partie gauche de la fenêtre d'édition. En double-cliquant sur un des deux cercles inférieurs, vous ouvrez à nouveau la boîte de dialogue de configuration. Entrez les variables suivantes dans le champ **Changer** du cercle associé:

pour le cercle du milieu: L1_yellow

pour le cercle inférieur: L1_green

Sélectionnez maintenant la couleur adéquate pour chaque cercle (jaune ou vert), dans la catégorie **Couleurs** et dans la zone **Couleur d'alarme**.

Le boîtier des feux

Sélectionnez à présent la commande 'Insérer' + "**Rectangle**", et insérez un rectangle qui entoure les trois cercles, en procédant de la même façon que précédemment pour le premier cercle. Choisissez pour le rectangle à nouveau une couleur aussi neutre que possible et activez la commande 'Extras' + "Arrière-plan", pour que les cercles deviennent à nouveau visibles.

Au cas où le mode simulationn'est pas encore activé, vous pouvez l'activer au moyen de la commande 'En Ligne' + `Simulation'.

Si vous lancez à présent la simulation au moyen des commandes 'En Ligne' + `Accéder au système' et 'En Ligne' + `Démarrer', vous pouvez suivre le changement de couleur du premier feu.

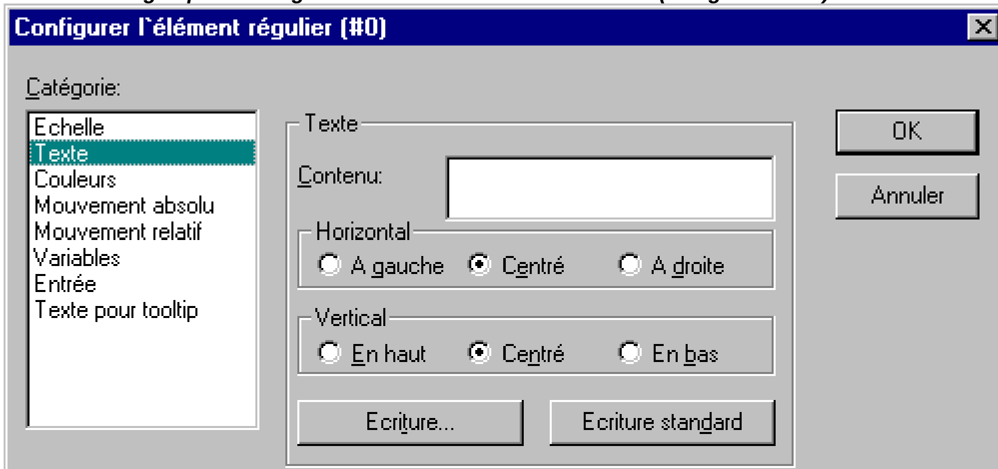
Le deuxième feu

Pour créer le deuxième feu, le plus simple est de copier l'ensemble des composants du premier feu. Pour ce faire, marquez tous les éléments du premier feu et copiez-les (comme précédemment pour les lampes du premier feu) au moyen des commandes 'Editer' + `Copier' et 'Editer' + `Coller'. Pour terminer la visualisation du deuxième feu, vous n'avez plus qu'à transformer le texte "S1" en "S2" dans les boîtes de dialogue de visualisation respectives.

L'interrupteur ON

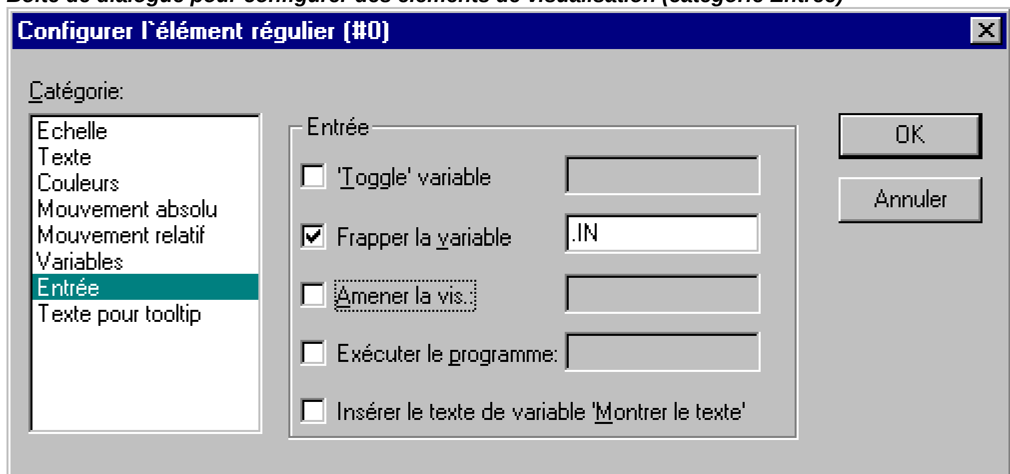
Insérez un rectangle, attribuez les couleurs de votre choix, selon ce qui est décrit ci-dessus dans le cas des feux, et entrez .ON dans **Variables** si vous souhaitez un **Changer la couleur**. A l'intérieur de la catégorie **Texte**, entrez "MARCHE" dans le champ **Contenu**.

Boîte de dialogue pour configurer des éléments de visualisation (catégorie Texte)



Pour pouvoir affecter à l'aide d'un clic de la souris la valeur TRUE à la variable ON, vous devez activer l'option **Frapper la variable** dans la catégorie **Entrée** et ensuite entrer la variable .ON. Sélectionnez en plus l'option **Variable à impulsion** et entrez à cet endroit la variable .ON. Si cette option est activée, la valeur TRUE est affectée à la variable .ON en cliquant avec la souris sur l'élément de visualisation, alors que si vous relâchez la touche de la souris, la valeur FALSE est réaffectée à cette variable (ainsi, nous avons créé un interrupteur simple pour les besoins de notre programme de feux).

Boîte de dialogue pour configurer des éléments de visualisation (catégorie Entrée)

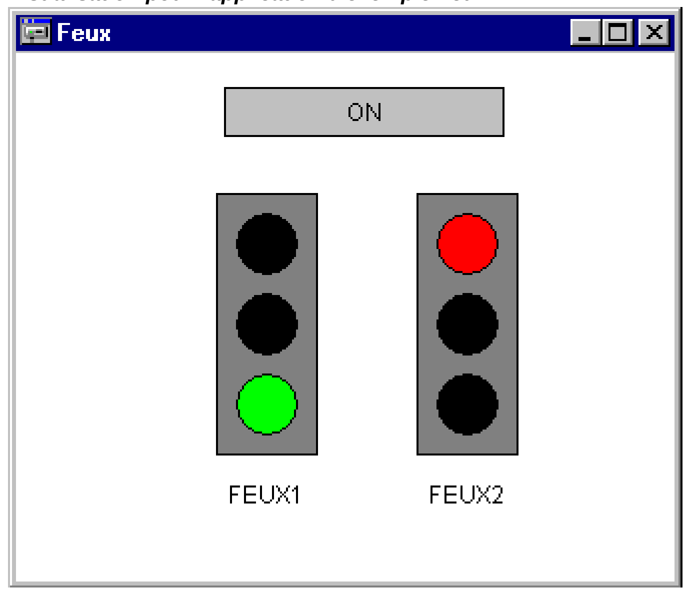


Inscription dans la visualisation

Pour compléter la visualisation, il convient d'ajouter deux rectangles aplatis et de les placer en dessous des feux.

Au niveau de la boîte de dialogue de visualisation, dans la catégorie Couleurs et dans **Cadre**, vous choisissez à chaque fois la configuration 'Aucune couleur de cadre'. Dans la catégorie **Texte** et dans le champ **Contenu** vous entrez respectivement "Feux 1" et "Feux 2". Votre visualisation ressemble maintenant à ceci:

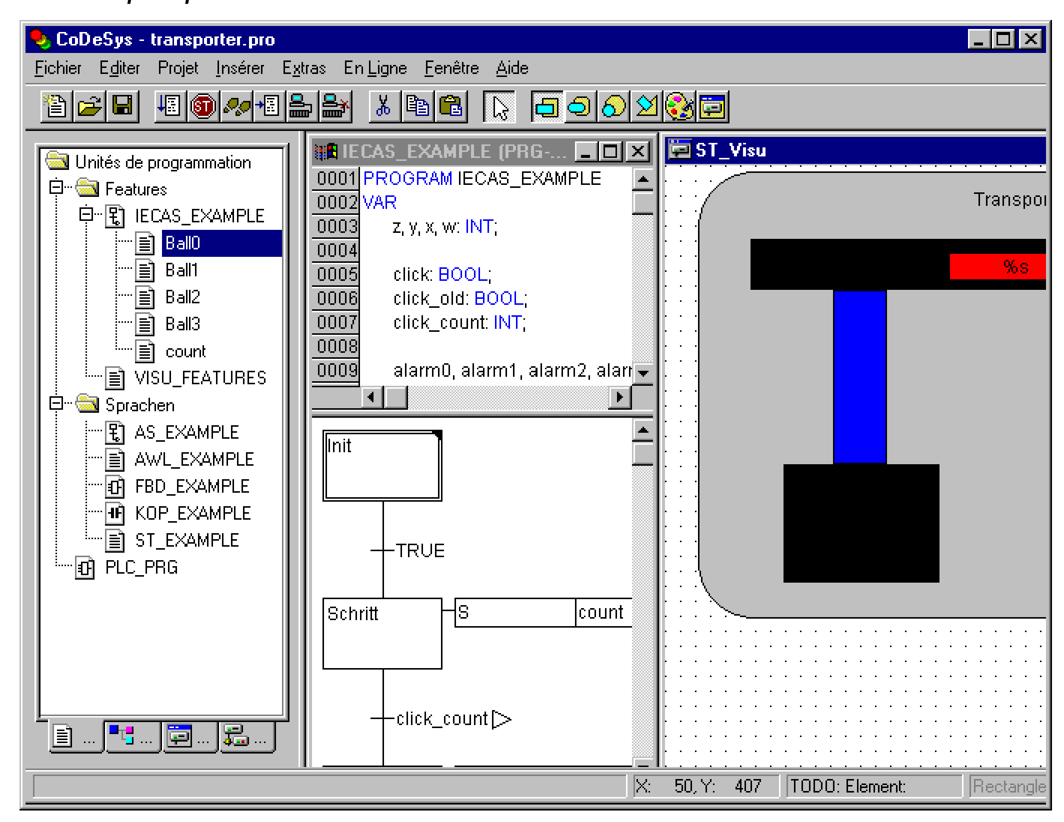
Visualisation pour l'application d'exemple Feux



4 Les composants dans le détail

4.1 La fenêtre principale

La fenêtre principale



Les éléments suivants sont situés dans la fenêtre principale de CoDeSys (de haut en bas):

- La barre de menus (Un grand nombre de commandes de menus se trouve également dans le menu contextuel, ce dernier s'ouvrant par le biais du bouton droit de la souris)
- La barre d'outils (facultative); avec des boutons pour exécuter plus rapidement les commandes de menus.
- L'Organisateur d'objets avec des onglets pour les modules, types de données, visualisations et ressources**.**
- Une barre de fractionnement verticale entre l'Organisateur d'objets et l'environnement de travail de CoDeSys.
- L'environnement de travail qui comprend les fenêtres des éditeurs.
- La fenêtre de messages (facultative).
- La barre d'état (facultative); avec des informations sur l'état actuel du projet.

Barre de menus

La barre de menus est située au niveau du bord supérieur de la fenêtre principale. Elle contient toutes les commandes des menus.



Barre d'outils

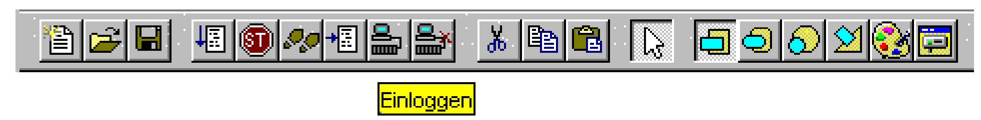
En cliquant avec la souris sur une icône de la barre d'outils, il est possible de sélectionner plus rapidement une commande de menu. La sélection d'icônes s'adapte automatiquement à la fenêtre active.

La commande n'est exécutée qu'en enfonçant et en relâchant la touche de la souris préalablement positionnée sur l'icône.

Si vous maintenez pendant un temps réduit le pointeur de la souris sur une icône de la barre d'outils, le nom de l'icône est affiché dans une info-bulle.

Pour recevoir la description de chaque icône de la barre d'outils il suffit de sélectionner dans l'aide l'éditeur à propos duquel vous souhaitez recevoir des informations et de cliquer sur l'icône de la barre d'outils qui vous intéresse.

L'affichage de la barre d'outils est facultatif (voir 'Projet' + 'Options', Catégorie "Environnement de travail").



Organisateur d'objets

L'Organisateur d'objets se situe toujours sur la partie gauche de CoDeSys. Dans la partie inférieure se trouvent quatre onglets avec des symboles pour les quatre types d'objets, à savoir les modules , **types de données** **visualisations** et **ressources** . Pour passer d'un type d'objet à un autre, cliquez sur l'onglet approprié avec la souris ou bien utilisez la touche directionnelle gauche ou droite.

Des symboles supplémentaires avant ou après les entrées d'objets caractérisent certains statuts concernant les Fonctions En Ligne et la Connexion ENI vers une base de données.

Le chapitre 4.4, 'Gestion des objets', vous indiquera comment travailler avec les objets dans l'Organisateur d'objets.

Barre de fractionnement

La barre de fractionnement constitue la frontière entre deux fenêtres accolées. CoDeSys prévoit des barres de fractionnement entre l'Organisateur d'objets et l'environnement de travail de la fenêtre principale, entre l'interface (partie de déclaration) et l'implémentation (partie instructions) des modules et entre l'environnement de travail et la fenêtre de messages.

Si vous amenez le pointeur de la souris sur la barre de fractionnement, vous pouvez déplacer cette dernière. Pour ce faire, déplacez la souris en maintenant le bouton gauche de la souris enfoncé.

Tenez compte du fait que la barre de fractionnement conserve toujours sa position absolue, même si la taille de la fenêtre est modifiée. Si la barre de fractionnement semble avoir disparue, alors il convient tout simplement d'agrandir votre fenêtre.

Environnement de travail

L'environnement de travail se trouve dans la partie droite de la fenêtre principale de CoDeSys. Tous les éditeurs relatifs aux objets et à la gestion des bibliothèques s'ouvrent au niveau de cette zone.

Vous trouvez la description des éditeurs aux chapitres 5, Les éditeurs dans CoDeSys, 6, Les ressources, Manuel CoDeSys Visualisation et chapitre 6.4, Gestion des bibliothèques. Le nom de chaque objet apparaît dans la barre de titre de la fenêtre, et une abréviation y est ajoutée entre parenthèses dans le cas des modules, indiquant le type de module et le langage de programmation utilisé.

Dans l'option "Fenêtre" se trouvent toutes les commandes relatives à la gestion des fenêtres.

Fenêtre de messages

La fenêtre de messages se situe dans la fenêtre principale, en dessous de l'environnement de travail et est séparée de celui-ci par une barre de fractionnement.

Elle fournit tous les messages relatifs aux derniers procédés de compilation, de vérification et de comparaison effectués. Les résultats de recherches et les 'Listes des références croisées' peuvent également être affichés ici.

Si, dans la fenêtre de messages, vous exécutez avec la souris un double-clic sur un message ou si vous enfoncez la touche <Entrée>, alors l'éditeur de l'objet s'ouvre. La ligne concernée de l'objet est marquée. Les commandes 'Editer' + "Erreur prochaine" et 'Editer' + "Erreur précédente" permettent de se déplacer rapidement d'un message d'erreur à l'autre.

L'affichage de la fenêtre de messages est facultative.

Barre d'état

La barre d'état, qui se situe au niveau du cadre inférieur de la fenêtre principale de CoDeSys, vous indique des informations sur le projet actuel et sur les commandes de menu.

Si un énoncé est valable, alors l'énoncé apparaît à droite dans la barre d'état, dans une couleur de police noire, sinon il apparaît dans une couleur de police grise.

Si vous travaillez en ligne, alors l'expression En Ligne apparaît dans une couleur de police noire. Si en revanche vous travaillez hors ligne, cette expression apparaît dans une couleur de police grise.

Lorsque vous êtes en mode En Ligne, la barre d'état vous renseigne si vous vous trouvez dans la simulation (**SIM**), si le programme est exécuté (**EXECUTE**), si un point d'arrêt est défini (**PT ARRET**) ou si des valeurs de variables sont forcées (**FORCER**).

Dans le cas d'éditeurs littéraux, le numéro de ligne et de colonne de la position actuelle du curseur est indiquée (p.ex. **L.:5, Col.:11**).

Si le pointeur de la souris se situe dans une visualisation, la **position** actuelle **X** et **Y** du curseur par rapport au coin supérieur gauche de l'image est indiquée, en pixel. Si le pointeur de la souris se situe sur un **élément** ou si un élément est en cours de traitement, le numéro de celui-ci est indiqué. Si vous avez choisi un élément à insérer, cela apparaît également (p.ex. **Rectangle**).

Si vous avez sélectionné une commande de menu, sans l'avoir encore activée, alors une brève description apparaît dans la barre d'état.

L'affichage de la barre d'état est facultatif (voir 'Projet' + "Options", Catégorie "Environnement de travail").

Menu contextuel

Raccourci : <Maj>+<F10>

Au lieu d'utiliser la barre de menus pour exécuter une commande, vous pouvez utiliser le bouton droit de la souris. Le menu qui est alors affiché contient les commandes utilisées les plus fréquemment en rapport avec un objet marqué ou avec l'éditeur actif. La sélection des commandes disponibles s'adapte automatiquement à la fenêtre active.

4.2 Projet Options

Les paramètres réglés sous 'Projet' 'Options' servent entre autres à la configuration de l'affichage de la fenêtre principale CoDeSys. À moins que ce ne soit stipulé autrement, ils sont mémorisés dans le fichier "Codesys.ini" et sont restitués lors du prochain démarrage de CoDeSys.

Une représentation des options de projet configurées pour le projet se trouve dans les ressources au niveau du l'Environnement de travail.

On accède à la boîte de dialogue 'Options' au moyen de cette commande. Les possibilités de paramétrage sont réparties en différentes catégories. Sélectionnez dans la partie gauche de la boîte

de dialogue la catégorie voulue au moyen d'un clic avec la souris ou à l'aide des touches directionnelles et modifiez les options dans la partie droite.

Voici les catégories :

Ouvrir & Fermer

Informations utilisateur

Editeur

Environnement de travail

Couleurs

Répertoires

Journal

Options de compilation

Mots de passe

Download Sourcecode

Configuration de symbole

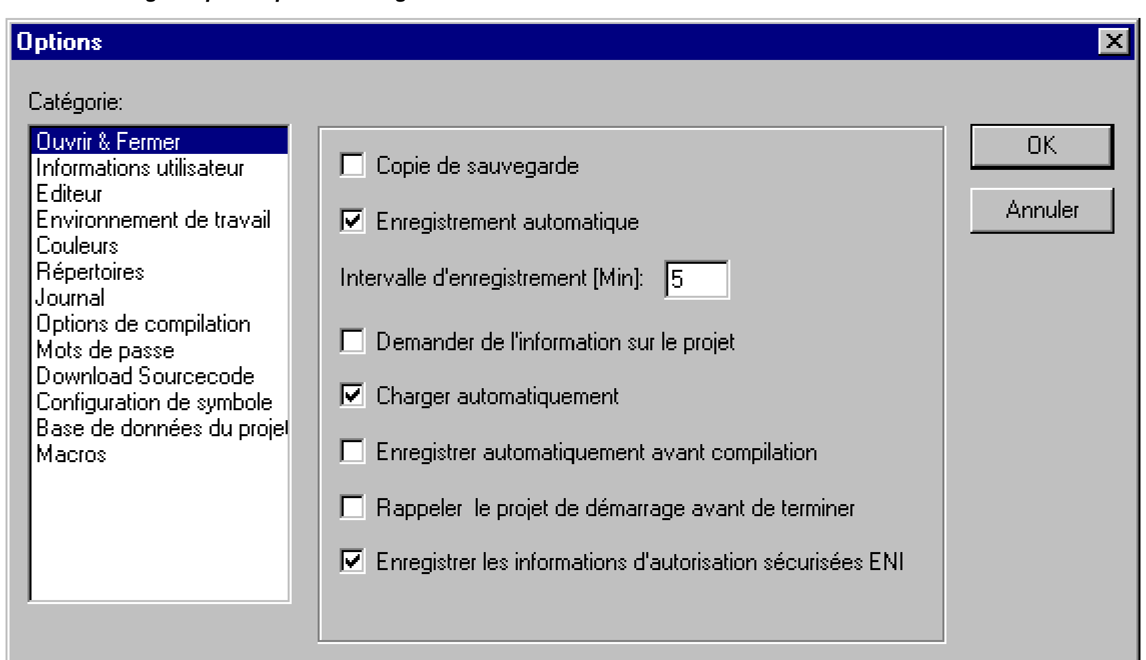
Base de données du projet

Macros

Ouvrir & Fermer

Si vous sélectionnez cette catégorie dans la boîte de dialogue Options, vous obtenez la boîte de dialogue suivante:

Boîte de dialogue Options pour la catégorie Ouvrir & Fermer

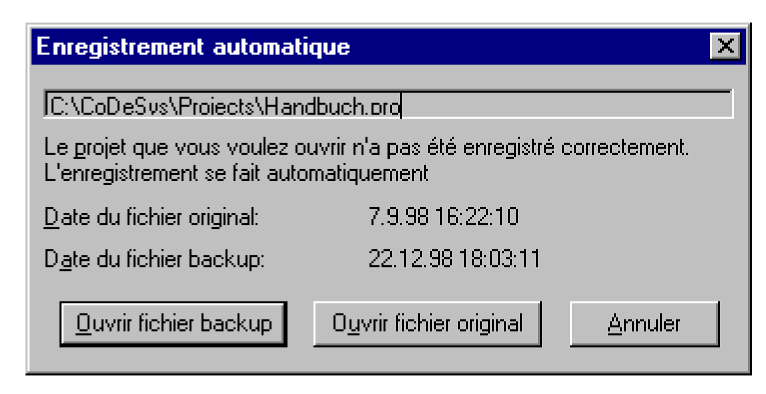


Lorsqu'une option est activée, un crochet (✓) apparaît devant l'option.

Copie de sauvegarde: CoDeSys sauvegarde l'ancien fichier lors de chaque enregistrement par la commande 'Fichier' 'Enregistrer' dans un fichier de sauvegarde, avec l'extension ".bak". Contrairement au fichier de sauvegarde *.asd (voir ci-dessous, 'Enregistrement automatique'), ce fichier est gardé même après avoir terminé le programme. Ainsi, vous pouvez toujours restituer la version qui précède le dernier enregistrement.

Enregistrement automatique: Le projet ouvert est fréquemment sauvegardé selon un intervalle que vous aurez fixé (**Intervalle d'enregistrement)** dans un fichier temporaire avec l'extension ".asd". Ce

fichier est effacé lorsque le programme s'est terminé normalement. Si, pour une raison ou une autre, CoDeSys ne se termine pas "normalement" (p.ex. lors d'une coupure de courant), le fichier n'est pas effacé. Lorsque, dans ce cas, vous ouvrez à nouveau le projet, le message suivant apparaît:



Vous pouvez alors décider si vous ouvrez le fichier original ou le fichier de sauvegarde.

Demander de l'information sur le projet: alors la boîte de dialogue **Info sur le projet** s'ouvre automatiquement lors de l'enregistrement d'un nouveau projet ou lors de la l'enregistrement d'un projet sous un nouveau nom. Vous pouvez afficher et modifier les informations sur le projet au moyen de la commande 'Projet''Info projet'.

Charger automatiquement: alors le dernier projet ouvert sera chargé automatiquement lors du prochain lancement de **CoDeSys.** Le chargement d'un projet lors du lancement de **CoDeSys** peut également s'effectuer en spécifiant un projet dans la ligne de commande.

Enregistrer avant la compilation: Le projet est enregistré avant toute compilation. Un fichier est alors créé avec l'extension ".asd", et celui-ci se comporte comme avec l'option 'Enregistrement automatique' décrite ci-dessus.

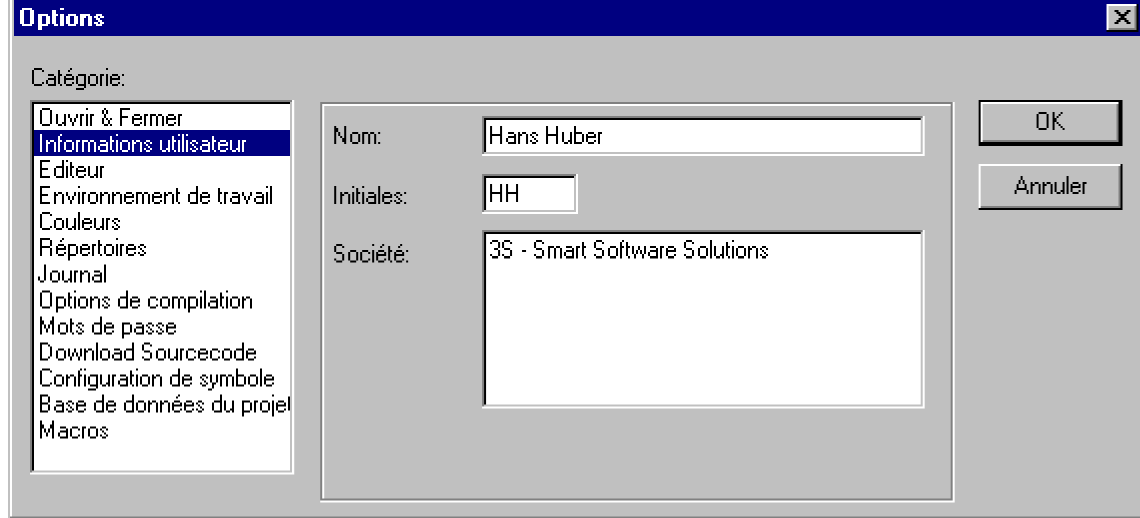
Rappeler le projet de démarrage avant de terminer: Si, depuis la création du projet d'initialisation, le projet sous forme modifiée a été chargé sur l'automate sans avoir auparavant créé un nouveau projet d'initialisation, ceci sera signalé à l'utilisateur lorsqu'il tentera de quitter le projet : "Aucun projet d'initialisation n'a été créé depuis le dernier téléchargement. Voulez-vous malgré tout terminer?"

Enregistrer les informations d'autorisation sécurisées ENI: Le nom d'utilisateur et le mot de passe tels qu'ils ont été introduits le cas échéant pour l'accès à la base de données ENI, sont enregistrés avec le projet.

Informations utilisateur

Si vous sélectionnez cette catégorie, vous obtenez la boîte de dialogue suivante:

Boîte de dialogue Options pour la catégorie Informations utilisateur

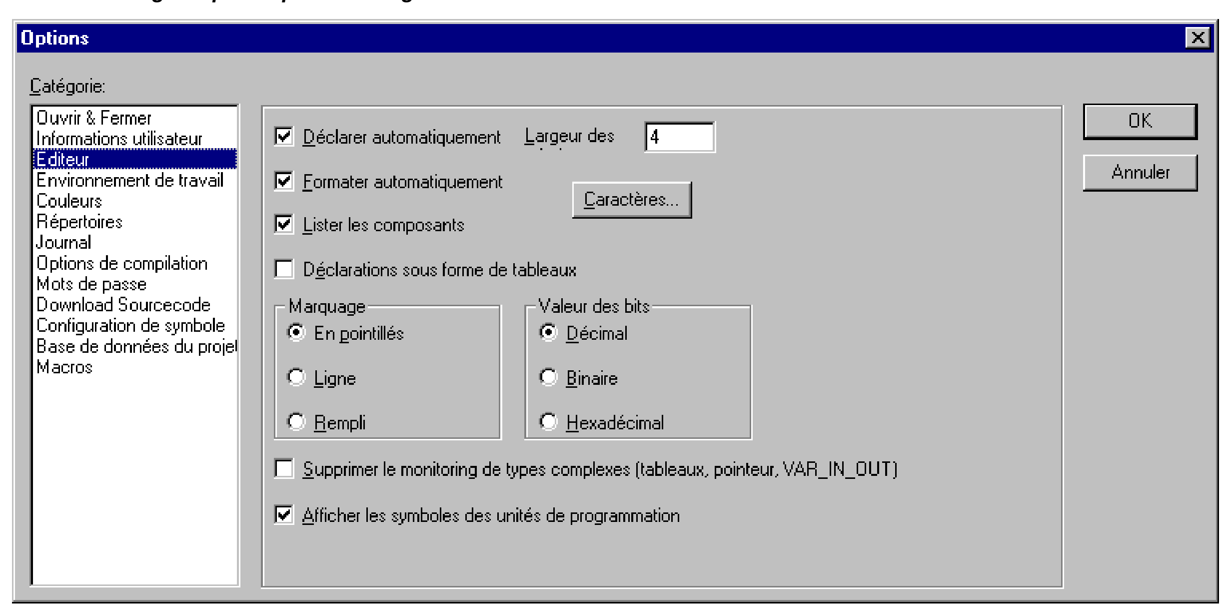


Les informations utilisateur comprennent le **Nom** de l'utilisateur, ses **Initiales** et la **Société** pour laquelle il travaille. Chacune des saisies peut être modifiée et est enregistrée avec le projet.

Editeur

Si vous sélectionnez cette catégorie, vous obtenez la boîte de dialogue suivante:

Boîte de dialogue Options pour la catégorie Editeur



Lorsqu'une option est activée, un crochet (✓) apparaît devant l'option.

Déclarer automatiquement: Si cette option a été sélectionnée, alors apparaît dans chaque éditeur, après entrée d'une variable non encore déclarée, une boîte de dialogue qui aide à déclarer cette variable.

Formater automatiquement: Si cette option a été sélectionnée, **CoDeSys** réalise un formatage automatique dans l'éditeur de Liste d'Instructions et dans l'éditeur de déclaration. A l'issue de chaque ligne, les formatages suivants sont effectuées:

- Les opérateurs qui figurent en minuscules sont visualisés en majuscules
- Des tabulateurs sont insérés, de sorte qu'il en résulte une répartition uniforme des colonnes.

Lister les composants: Si cette option est activée, vous disposez de la fonction **Intellisense** dans CoDeSys. Si vous n'introduisez qu'un point aux endroits où un identificateur doit être saisi, vous obtenez une liste de sélection reprenant toutes les variables disponibles pour cet endroit. Cette fonction Intellisense est disponible, non seulement dans l'éditeur, mais aussi dans le gestionnaire d'espion et de recettes, dans la visualisation et dans la configuration de l'histogramme.

Déclarations sous forme de tableau: Si cette option a été sélectionnée, vous pouvez éditer des variables sous forme de tableau au lieu de le faire au moyen de l'éditeur de déclaration classique. Ce tableau est ordonné de la même façon qu'une cartothèque, dans lequel se trouvent des onglets pour les variables d'entrée, de sortie, les variables locales et les variables d'entrée-sortie. Vous disposez pour chaque variable des champs **Nom, Adresse, Type, Initial, Commentaire**.

Largeur de tabulation: Ici vous pouvez spécifier la largeur d'une tabulation visualisée dans les éditeurs. La largeur configurée par défaut est de 4 caractères, avec une largeur de caractère qui dépend à son tour de la configuration de la police de caractères.

Caractères: En appuyant sur ce bouton vous pouvez sélectionner la police dans tous les éditeurs **CoDeSys**. La taille de la police est l'unité de base pour toutes les opérations de dessin. Le choix d'une taille de police plus grande agrandit ainsi la représentation et même l'impression pour chaque éditeur de **CoDeSys**.

Après avoir sélectionné la commande, la boîte de dialogue pour la sélection de la police de caractère, du style et de la taille de police s'ouvre.

Boîte de dialogue de configuration de la police

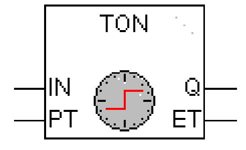
Schriftart
Schriftart: MS Sans Serif
MS Sans Serif
MS Serif
MS SystemEx
Playbill
Small Fonts
System
Times New Roman
Schriftschnitt: Fett
Standard
Kursiv
Fett
Fett Kursiv
Grad: 8
8
10
12
14
18
24
OK
Abbrechen
Muster
AaBbYyZz
Schrift:
Westlich

Marquage: Ici vous pouvez choisir si une sélection dans les éditeurs graphiques doit être représentée au moyen d'un rectangle en pointillés (**En pointillés**), d'un rectangle en ligne continue (**Ligne**) ou d'un rectangle plein (**Rempli**). Dans le dernier cas, la sélection est visualisée "en négatif" (inverse).

Valeur des bits: Ici vous pouvez choisir si des informations binaires (de type BYTE, WORD, DWORD) doivent être visualisées, au niveau de l'espionnage, en mode **Décimal**, **Hexadécimal** ou **Binaire**.

Supprimer le monitoring de types complexes (tableaux, pointeur, VAR_IN_OUT): Si cette option est activée, les types de données complexes comme les tableaux, les pointeurs, VAR_IN_OUTs ne seront pas représentés dans la fenêtre d'espion du mode En ligne.

Afficher les symboles des unités de programmation: Si cette option est activée, des symboles sont représentés dans les boîtiers de modules, pour autant qu'ils soient disponibles dans la bibliothèque des répertoires sous forme d'images bitmap. Le nom du fichier bitmap doit se composer du nom du module et de l'extension .bmp. Exemple : pour le module TON, le symbole suivant est contenu dans le fichier TON.bmp :



Environnement de travail

Si vous sélectionnez cette catégorie, vous obtenez la boîte de dialogue suivante:

Boîte de dialogue Options pour la catégorie Environnement de travail

Options
Catégorie:
Ouvrir & Fermer
Informations utilisateur
Editeur
Environnement de travail
Couleurs
Répertoires
Journal
Options de compilation
Mots de passe
Download Sourcecode
Configuration de symbole
Base de données du projet
Macros
Barre d'icônes
Afficher la zone d'impression
Représentation MDI
Ligne d'état
F4 ignore les avertissements
Opération en ligne en mode sécurité
Faire une requête des paramètres de communication avant le login
Ne pas enregistrer les paramètres de communication dans le projet
Langue: Français
OK
Annuler

Barre d'icônes: La barre d'outils avec les boutons pour sélectionner plus rapidement les commandes de menu est rendue visible en dessous de la barre de menus.

Ligne d'état: La barre d'état est rendue visible au niveau du bord inférieur de la fenêtre principale de **CoDeSys**.

Opération en ligne en mode sécurité: Une boîte de dialogue avec une demande de sécurité apparaît en mode **En Ligne**, pour les commandes Démarrer, Arrêter, Reset, Point d'arrêt actif, Cycle indépendant, Ecrire valeurs des variables, Forcer valeurs des variables, Arrêter de forcer, pour demander si la commande doit réellement être exécutée. Si le système d'exécution le supporte, un dialogue élargi apparaît lors du chargement du projet sur l'automate programmable : Ce dialogue indique en outre les informations d'un projet se trouvant éventuellement déjà sur l'automate ainsi que celles du nouveau projet à charger. Ces informations de projet apparaissent également lors de la création d'un projet d'initialisation, si un tel projet est déjà présent sur l'automate programmable. Cette option est enregistrée avec le projet.

Faire une requête des paramètres de communication avant le login: Suite à la commande 'En Ligne' 'Login', la boîte de dialogue pour les paramètres de communication apparaît à l'écran. On ne passe au mode En ligne que lorsque cette boîte de dialogue a été fermée à l'aide de OK.

Ne pas enregistrer les paramètres de communication dans le projet: Les paramètres de la boîte de dialogue pour les paramètres de communication ('En Ligne' 'Paramètre de communication') ne sont pas enregistrés dans le projet.

Afficher la zone d'impression: Les limites de la zone d'impression en cours de réglage sont marquées par des traits discontinus rouges dans chaque fenêtre d'éditeur. L'importance de cette zone dépend des caractéristiques de l'imprimante (format de papier, alignement) et de la grandeur de la zone "Contenu" de la mise en page réglée (Menu 'Fichier' 'Paramètres documentation').

F4 ignore les avertissements: En appuyant sur F4 dans la fenêtre des messages après une compilation, l'accent est mis sur les lignes comprenant les messages d'erreur, et les avertissements sont ignorés.

Représentation MDI: Cette option (**M**ultiple-**D**ocument-**I**nterface) est activée par défaut, rendant ainsi possible l'ouverture simultanée de plusieurs objets (fenêtres). Si cette option est désactivée (mode SDI, Single-Document Interface), on ne peut alors ouvrir qu'une fenêtre à la fois dans l'environnement de travail, et cette fenêtre est représentée en mode plein écran. Exception : l'action d'un programme peut être représentée aussi en MDI en même temps que le programme lui-même.

Langue:Sélectionnez dans quelle langue les textes des menus et des boîtes de dialogue ainsi que l'aide En ligne doivent être affichés.

Remarque : Notez que la sélection d'une langue n'est pas possible sous Windows 98 !

Couleurs

Si vous sélectionnez cette catégorie, vous obtenez la boîte de dialogue suivante:

Boîte de dialogue Options pour la catégorie Couleurs

Options

Catégorie:

Ouvrir & Fermer
Informations utilisateur
Editeur
Environnement de travail
Couleurs
Répertoires
Journal
Options de compilation
Mots de passe
Download Sourcecode
Configuration de symbole
Base de données du projet
Macros

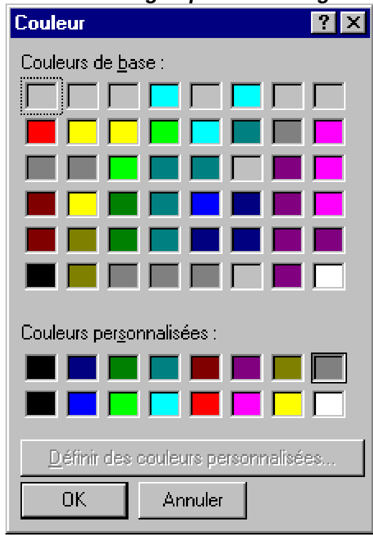
Numéros de ligne... | Position actuelle...
Positions des points d'arrêt... | Position parcourue...
Point d'arrêt défini... | Monitoring de BOOL...

OK
Annuler

Vous pouvez éditer la configuration par défaut des couleurs de **CoDeSys**. Vous pouvez choisir de modifier la configuration des couleurs pour les **Numéros de ligne** (configuration par défaut: gris clair), pour la **Positions des points d'arrêt** (gris foncé), pour un **Point d'arrêt défini** (bleu clair), pour la **Position actuelle** (rouge), pour les **Positions parcourues** (vert) ou pour **Monitoring de BOOL** de valeurs de type booléen (bleu).

Si vous avez sélectionné un des boutons, la boîte de dialogue pour entrer des couleurs s'ouvre.

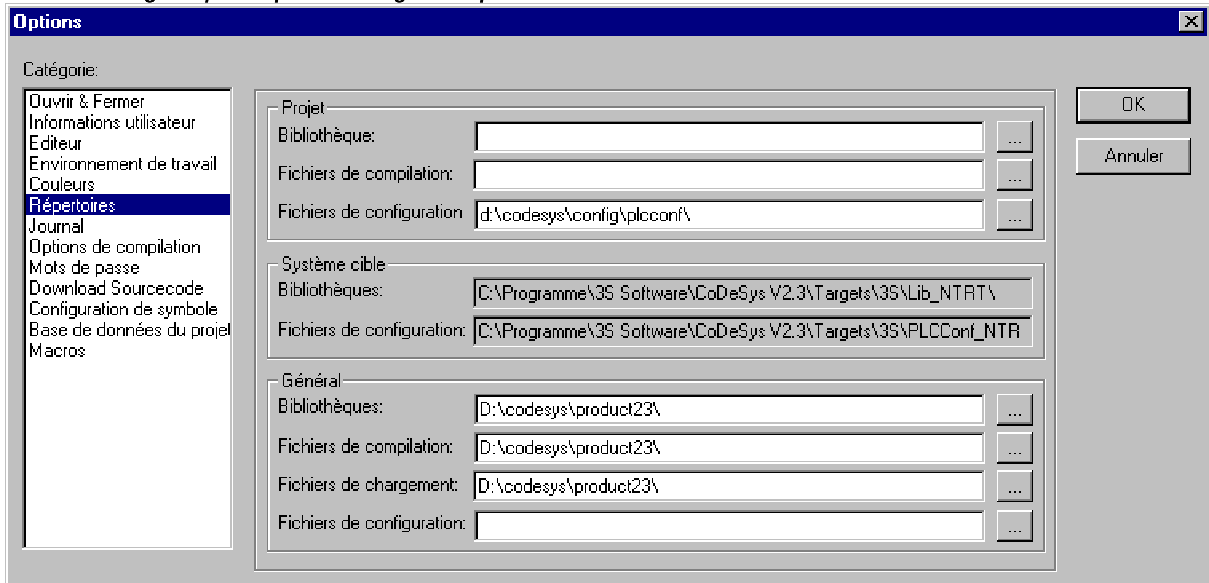
Boîte de dialogue pour la configuration des couleurs



Répertoires

Si vous sélectionnez cette catégorie, vous obtenez la boîte de dialogue suivante:

Boîte de dialogue Options pour la catégorie Répertoires



Vous pouvez introduire, dans les zones **Projet** et **Général,** des répertoires que **CoDeSys** devra parcourir à la recherche de **bibliothèques** et de**fichiers de configuration** d'automate ou qu'il devra utiliser pour la réception de fichiers de **compilation** ou de fichiers source de **téléchargement en amont**. Si vous activez le bouton (…) derrière un champ, alors la boîte de dialogue pour sélectionner un répertoire s'ouvre. Plusieurs chemins d'accès séparés par un point-virgule ";" peuvent être introduits pour les fichiers de bibliothèque et de configuration.

Remarque: Les chemins d'accès aux bibliothèques peuvent être saisis en relation au répertoire de projet actuel, en les faisant précéder d'un ".". Si on saisit par exemple ".\libs", les bibliothèques seront également recherchées dans le répertoire 'C:\Programme\projects\libs', pour autant que le projet se trouve dans le répertoire 'C:\Programme\projects'.

Remarque: N'utilisez pas d'espaces ou de caractères spéciaux dans les chemins des répertoires, à l'exception de "_".

Les entrées dans la zone **Projet** sont enregistrées avec le projet. Les entrées dans la zone **Général**sont écrites dans le fichier Ini du système de programmation et valent pour tous les projets.

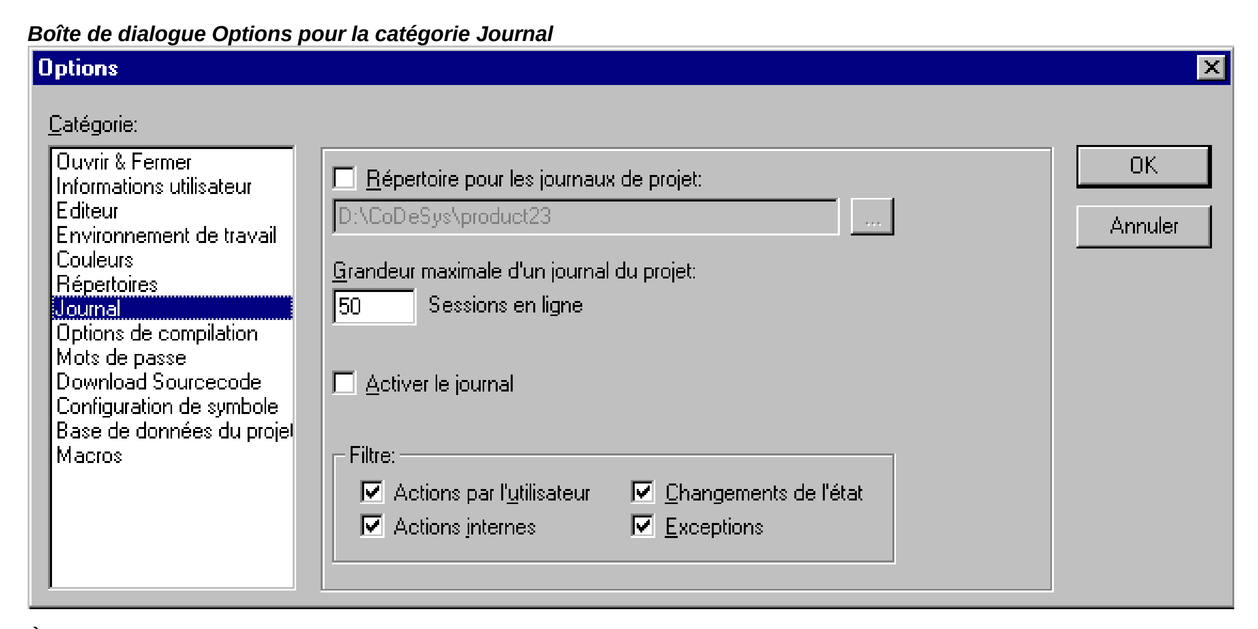
Les répertoires de bibliothèques et de fichiers de configuration, configurés dans le système cible, sont affichés dans la zone **Système cible** à l'aide, par exemple, des données dans le fichier cible. Vous ne pouvez pas éditer ces champs, mais une entrée peut être sélectionnée et copiée (menu contextuel bouton droit de la souris).

En règle générale, CoDeSys cherche d'abord parmi les répertoires classés sous 'Projet', ensuite les répertoires sous 'Système cible' et enfin ceux sous 'Général'. Si plusieurs fichiers portent le même nom, celui qui se trouve dans le premier répertoire parcouru est utilisé.

Journal

Si vous sélectionnez cette catégorie, vous obtenez la boîte de dialogue suivante :

Boîte de dialogue Options pour la catégorie Journal



À l'intérieur de cette boîte de dialogue, vous pouvez configurez un fichier qui, en tant que journal de projet, enregistre chronologiquement toutes les actions d'utilisateur et tous les processus internes durant le mode En ligne.

Si un projet existant est ouvert pour lequel aucun journal n'a été généré, une boîte de dialogue s'ouvre vous signalant qu'un journal est créé, qui ne mémorisera les entrées qu'à partir du prochain processus d'ouverture de session.

Lors de la sauvegarde du projet, le journal sera automatiquement enregistré sous forme de fichier binaire dans le répertoire de projet. Si vous souhaitez un autre répertoire cible, vous pouvez activer l'option **Répertoire pour les journaux de projet:** et introduire le chemin d'accès adéquat dans le champ d'édition. Par le biais du bouton , vous obtenez la boîte de dialogue 'Sélectionner le répertoire'.

Le fichier de journal enregistre automatiquement le nom du projet avec l'extension .log. Le nombre maximal de **Sessions en ligne** à enregistrer est fixé sous la **Grandeur maximale d'un journal du projet**. Si ce nombre est dépassé durant un enregistrement, les entrées les plus anciennes sont effacées au profit des nouvelles.

Cette fonction de journal est activée et désactivée par le biais du bouton d'option **Activer le journal**.

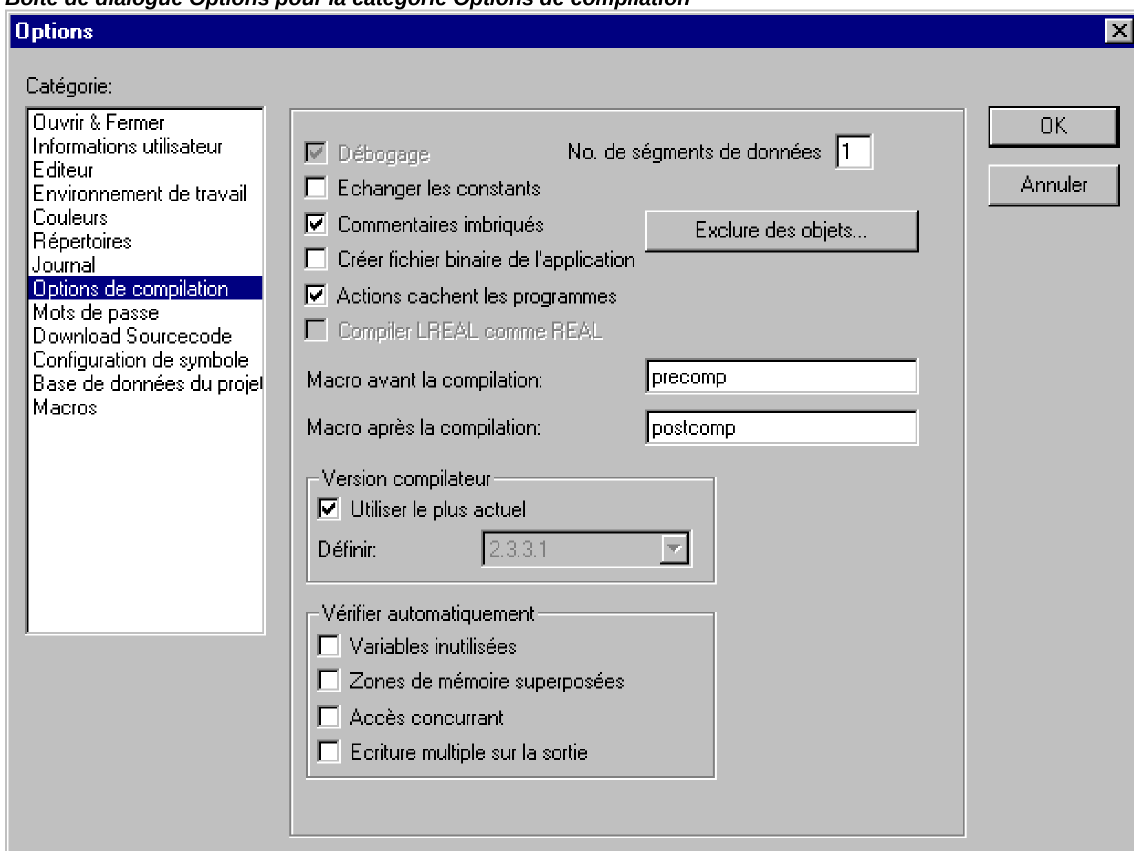
Dans la zone **Filtre**, vous pouvez sélectionner les actions qui doivent être enregistrées : Actions par l'utilisateur, Actions internes, changements de l'état, Exceptions. Les actions provenant de catégories munies ici d'un crochet seront seules affichées à la fenêtre du journal ou écrites dans le fichier du journal.

Vous pouvez ouvrir la fenêtre de journal par le biais de la commande 'Fenêtre' 'Journal'.

Options de compilation

Si vous sélectionnez cette catégorie, vous obtenez la boîte de dialogue suivante:

Boîte de dialogue Options pour la catégorie Options de compilation



Débogage:Selon le système cible, cette option peut être sélectionnée ou préréglée. Si elle est activée, le code peut être considérablement plus grand. Le choix de cette option permet de générer du code de débogage supplémentaire. Le code de débogage est nécessaire pour utiliser les fonctions de débogage disponibles sur CoDeSys. Si vous désactivez cette option, vous augmentez la rapidité d'exécution et vous autorisez des codes de taille plus petite. Cette option est enregistrée avec le projet.

Échanger constantes: Les constantes sont remplacées par leur valeur d'initialisation, au cours de la génération du code. Cette option permet d'accéder plus rapidement aux valeurs que dans le cas d'un accès via un emplacement de mémoire. Cependant les constantes ne peuvent pas être modifiées En Ligne dans ce cas ("**Forcer valeurs des variables**" et "**Ecrire valeurs des variables**").

Commentaires imbriqués : Vous pouvez introduire des commentaires imbriqués les uns dans les autres. Exemple :

```
(*
a:=inst.out; (* à vérifier *)
b:=b+1;
*)
```

Le commentaire débutant avec la première parenthèse ne se termine pas à la parenthèse située directement après 'vérifier', mais bien à la dernière parenthèse.

Créer fichier binaire de l'application: Une représentation binaire du code est créé (Projet d'initialisation) dans le répertoire de projet lors de la compilation. Nom de fichier: <nom de projet>.bin. Notez également à cet effet la possibilité de créer le projet d'initialisation ainsi que le fichier correspondant de total de contrôle En ligne sur l'automate (commande 'En Ligne' 'Créer projet d'initialisation), ou Hors ligne dans le répertoire de projet.

Actions cachent les programmes: Cette option est activée par défaut dès qu'un nouveau projet est créé. Sa signification : si une action locale porte le même nom qu'une variable ou un programme, l'exécution s'effectue dans l'ordre ci-après : variable locale, action locale, variable globale, programme.

Attention: Si on ouvre un projet qui a été créé avec une version précédente de CoDeSys, cette option est désactivée par défaut. Ainsi, on garde la hiérarchie qui a été appliquée lors de la création (variable locale, variable globale, programme, action locale).

Compiler LREAL comme REAL: Si cette option est activée (par défaut, elle ne l'est pas), les valeurs LREAL sont traitées comme des valeurs REAL lors de la compilation du projet. Cette option est utilisée afin de développer des projets indépendamment de la plate-forme.

En spécifiant le **Nombre de segments de données**, vous pouvez définir la quantité de segments de mémoire réservés aux données de votre projet dans l'automate programmable. Cet espace est nécessaire pour pouvoir effectuer un changement En ligne lorsque des nouvelles variables ont été introduites. Si en cours de compilation le message "Les variables globales occupent trop de mémoire." apparaît, augmentez le nombre de segments définis ici.

Exclure des objets: Ce bouton donne accès au dialogue **Exclure des objets de la compilation**: Sélectionnez au sein de l'arborescence affichée les modules du projet qui ne doivent pas être compilés lors d'une procédure de compilation, puis activez l'option **Ne pas compiler**. Les modules exclus de la compilation sont alors affichés en vert dans l'arborescence. Afin d'exclure automatiquement tous les modules qui ne sont pas utilisés au sein du projet, appuyez sur le bouton **Exclure inutilisés**.

Version compilateur: La version du compilateur à utiliser pour la compilation peut être définie ici. Les versions CoDeSys postérieures à V2.3.3 disposent de la version actuelle de compilateur ainsi que des versions précédentes jusqu'à V2.3.3 (pour chaque version / chaque Service Pack / chaque retouche). Si vous souhaitez qu'un projet soit toujours compilé avec la version de compilateur la plus récente, activez l'option **Utiliser le plus actuel**. Si le projet doit automatiquement être compilé avec une version précise de compilateur, définissez celle-ci dans le champ de sélection en regard de **Définir**.

Pour agir sur le processus de compilation, deux macros peuvent être utilisées :

La macro située dans le champ **Macro avant la compilation** est exécutée avant la compilation, la macro située dans le champ **Macro après la compilation** s'effectue après la compilation. Les commandes de macro ci-dessous ne peuvent cependant jamais être exécutées dans ce contexte : file new, file open, file close, file saveas, file quit, online, project compile, project check, project build, debug, watchlist.

Tous les paramètres fixés dans la boîte de dialogue des options de compilation sont enregistrés avec votre projet.

Vérifier automatiquement:

Afin de contrôler l'exactitude sémantique lors de chaque compilation du projet, vous pouvez activer les options ci-dessous :

- Variables inutilisées
- Zones de mémoire superposées
- Accès concurrent
- Ecriture multiple sur la sortie

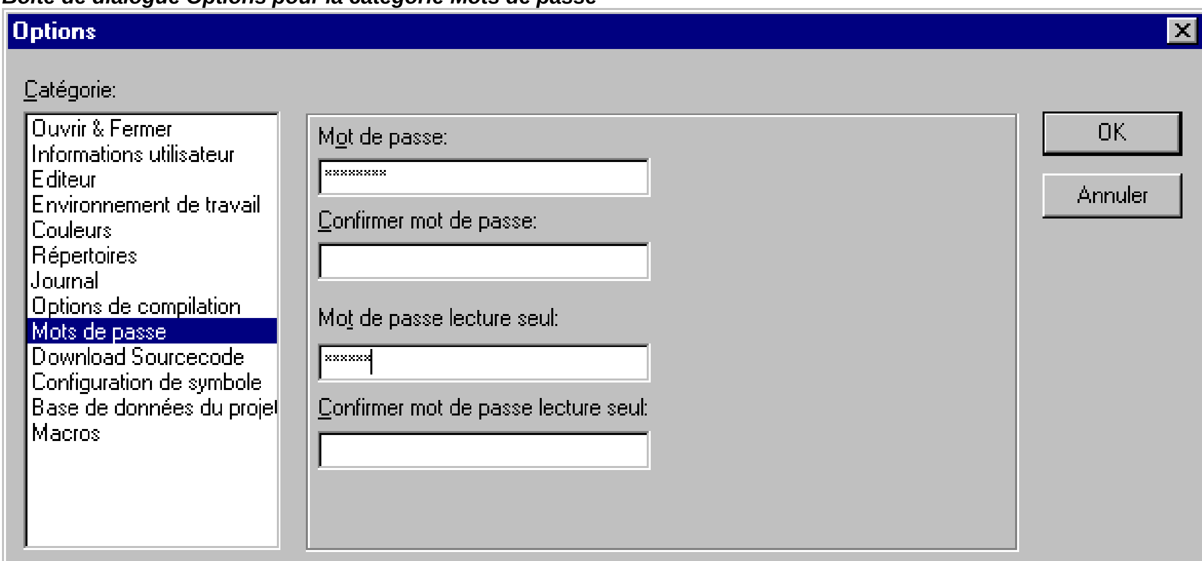
Les résultats seront affichés dans une fenêtre de messages. Ces contrôles peuvent être lancés de manière ciblée par le biais de l'instruction Vérifier dans le menu 'Projet'.

Remarque: Tous les réglages effectués au sein du dialogue des Options de compilation sont enregistrés en même temps que le projet.

Mots de passe

Si vous sélectionnez cette catégorie, vous obtenez la boîte de dialogue suivante:

Boîte de dialogue Options pour la catégorie Mots de passe



Pour protéger vos données contre des accès non autorisés, **CoDeSys** vous donne la possibilité de sécuriser l'ouverture et la modification de vos fichiers au moyen d'un mot de passe.

Entrez le mot de passe voulu dans le champ **Mot de passe**. Une étoile (*) apparaît à chaque fois que vous saisissez une lettre. Vous devez répéter ce même mot au niveau du champ **Confirmer mot de passe**. Fermer la boîte de dialogue par **OK**. Si le message

"Le mot de passe et la confirmation ne concordent pas."

apparaît, vous avez fait une faute de frappe au cours d'une des entrées. C'est pourquoi il est préférable de répéter les deux saisies jusqu'à ce que la boîte de dialogue se ferme sans qu'un message n'apparaisse.

Si maintenant vous ouvrez à nouveau le fichier après l'avoir enregistré, une boîte de dialogue apparaît pour vous demander d'entrer le mot de passe. Le projet ne s'ouvre que lorsque vous avez saisi le mot de passe correct. Si ce n'est pas le cas, le message suivant apparaît:

"Le mot de passe est incorrect."

Il vous est possible de protéger à l'aide d'un mot de passe non seulement l'ouverture mais aussi la modification d'un fichier. Pour ce faire, il faut effectuer une saisie dans le champ **Mot de passe lecture seule**, puis répéter cette saisie dans le champ du dessous.

Un projet protégé en écriture peut être ouvert sans mot de passe. Pour cela, appuyez simplement sur le bouton **Annuler**, si **CoDeSys** vous demande d'entrer le mot de passe lecture seule au moment d'ouvrir un fichier. Vous pouvez à présent compiler le projet, le charger dans l'automate programmable, procéder à une simulation, etc. Vous ne pouvez toutefois pas modifier le projet.

Il est important que vous mémorisiez bien les deux mots de passe, cela va de soi. Au cas où vous auriez réellement oublié un mot de passe, veuillez vous adresser à votre fabricant d'automate programmable.

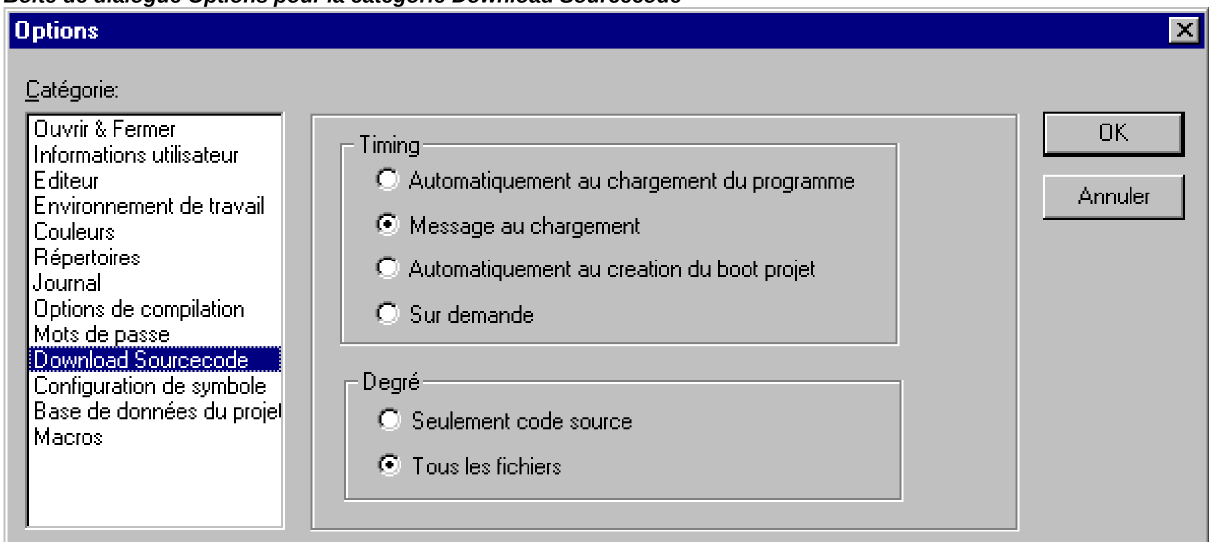
Les mots de passe sont enregistrés avec le projet.

Pour créer des droits d'accès spécifiques, vous pouvez définir des niveaux d'accès et 'Projet' 'Mots de passe pour niveau d'accès'.

Download Sourcecode

Si vous sélectionnez cette catégorie dans la boîte de dialogue Options, vous obtenez la boîte de dialogue suivante:

Boîte de dialogue Options pour la catégorie Download Sourcecode



Vous pouvez choisir le **Timing** et le **Degré** d'enregistrement du projet dans l'automate programmable. Afin d'être enregistrées, les données sont comprimées. L'option **Seulement code source** concerne uniquement le fichier CoDeSys (Zusatz.pro). L'option **Tous les fichiers** englobe également des fichiers supplémentaires, tels que les bibliothèques liées au projet, les visualisations sous forme d'image bitmap, les fichiers de configuration, etc.

Automatiquement au chargement du programme: Si cette option est activée, la commande 'En Ligne' + `Lancer' vous permet de charger automatiquement dans l'automate programmable les fichiers correspondant au degré choisi.

Message au chargement: Si cette option est activée, la commande 'En Ligne' + `Lancer' vous permet d'accéder à une boîte de dialogue avec l'interrogation suivante "Ecrire le code source dans l'automate programmable ?". Si vous appuyez sur **Oui**, les fichiers correspondant au degré sélectionné sont chargés automatiquement dans l'automate programmable. Si vous appuyez en revanche sur **Non**, la boîte de dialogue se referme.

Automatiquement à la création du projet d'initialisation : Si cette option est activée, la commande 'En Ligne' 'Créer projet d'initialisation' vous permet de charger automatiquement dans l'automate programmable les données choisies.

Sur demande: Si cette option est activée, les fichiers correspondant au degré choisi doivent être chargés exprès dans l'automate programmable à l'aide de la commande 'En Ligne' 'Application téléchargée du code source dans l'automate'

Le projet stocké dans l'automate programmable peut être à nouveau chargé dans l'ordinateur à l'aide de la commande '**Ouvrir projet de l'automate'** dans le menu 'Fichier' + "Ouvrir fichier". Les données sont décomprimées au moment du chargement.

Configuration de symbole

La boîte de dialogue donnée ici sert à la configuration du fichier de symboles. Celui-ci est créé sous forme de fichier de texte <Nom du projet>.sym ou de fichier binaire <Nom du projet>.sdb (en fonction de la gateway utilisée) dans le répertoire du projet. Le fichier de symboles est nécessaire pour l'échange de données avec l'automate programmable par le biais de l'interface de symboles, et est utilisé par exemple par le serveur gateway OPC ou DDE.

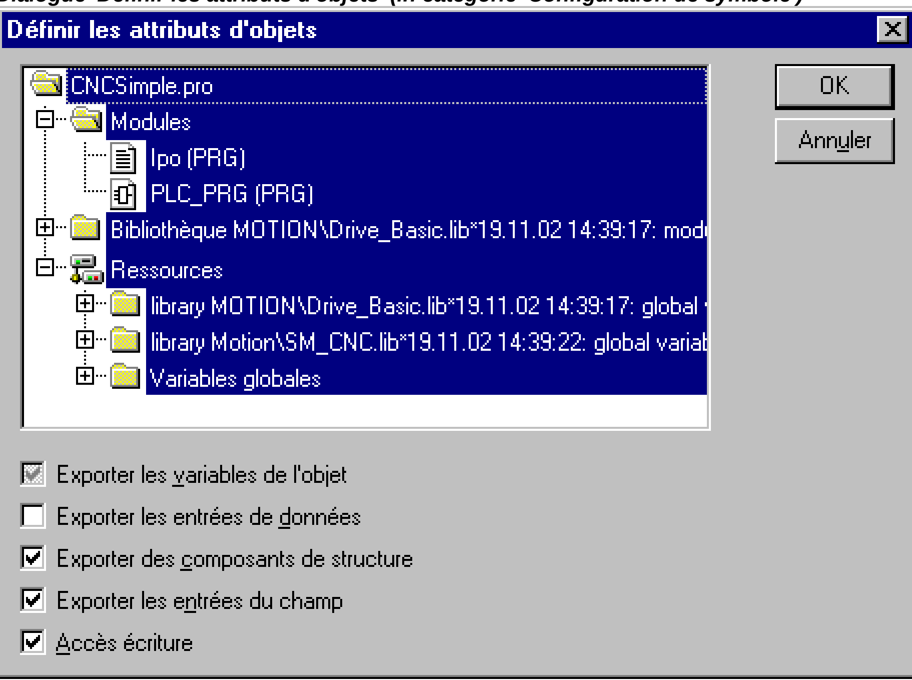
Si l'option **Générer des entrées de symboles** est sélectionnée, des entrées de symboles sont automatiquement créées dans le fichier des symboles à chaque compilation de projet.

Si en plus l'option **Générer un tableau XML** est activée, une version XML du fichier des symboles est en outre créée. Celle-ci est créée dans le répertoire du projet et est dénommée <Nom du projet>.SYM_XML.

Pour la configuration des entrées du fichier de symboles :

- Si l'option 'Configuration des symboles par fichier INI' est activée dans la configuration du système cible, la configuration des entrées de symboles est lue à partir de codesys.ini ou à partir d'un autre fichier INI nommé en cet endroit. (La boîte de dialogue CoDeSys 'Définir les attributs d'objets' n'est en aucun cas éditable.)
- Si l'option 'Configuration des symboles par fichier INI' n'est pas activée, les entrées de symboles sont créées selon les réglages que vous avez définis dans la boîte de dialogue 'Définir les attributs d'objets'. Vous y parvenez par le biais du bouton **Configurer un fichier de symbole** :

Dialogue 'Définir les attributs d'objets' (in catégorie 'Configuration de symbole')



Sélectionnez les modules de projet dans l'arborescence et activez les options souhaitées dans la partie inférieure de la boîte de dialogue en cliquant sur la (les) case(s) en regard. Les options activées sont munies d'un crochet. Les options suivantes sont possibles :

Exporter les variables de l'objet : Les variables de l'objet sélectionné sont reprises dans le fichier des symboles.

Les options suivantes ne valent que si l'option 'Exporter les variables de l'objet' est activée:

Exporter les entrées des données: Des entrées pouvant agir sur les variables globales sont créées pour les structures et tableaux de l'objet.

Exporter les composants de structure: Pour les structures de l'objet, une entrée propre est créée pour chaque composant de variable.

Exporter les composants de champ: Pour les tableaux de l'objet, une entrée propre est créée pour chaque composant de variable.

Accès écriture: Les variables de l'objet peuvent être modifiées à partir du serveur OPC.

Après que le réglage des options ait été effectué pour la sélection de module en cours, vous pouvez - sans devoir fermer la boîte de dialogue avec OK - sélectionner d'autres modules et leur attribuer une configuration d'options. Vous pouvez traiter autant de modules que vous le souhaitez les uns à la suite des autres.

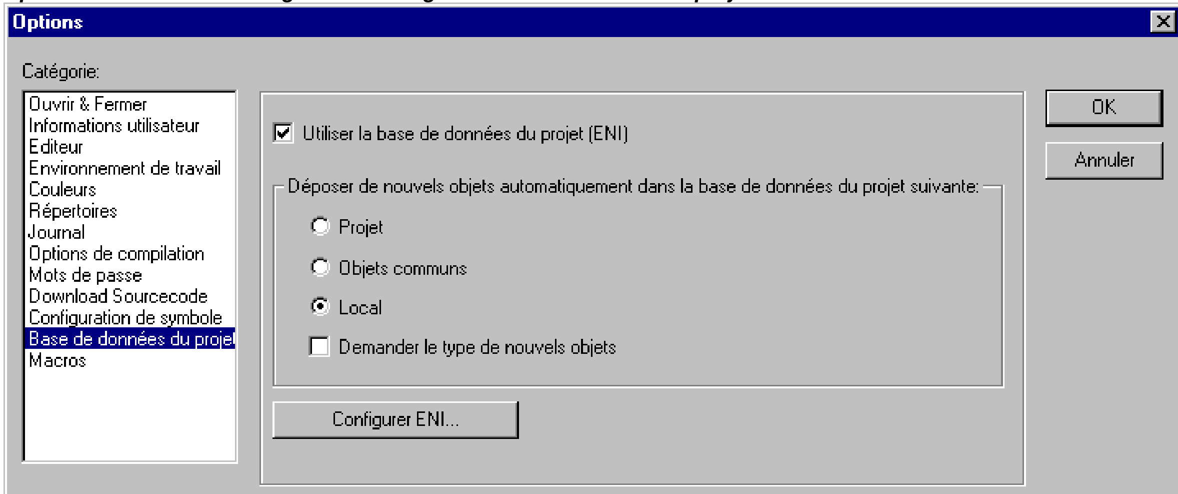
Lorsque vous fermez la boîte de dialogue à l'aide du bouton OK, toutes les configurations effectuées depuis l'ouverture de cette boîte sont enregistrées.

Remarque: Notez qu'il est possible, à l'aide de pragmas, de reprendre chaque variable de manière ciblée sans droit d'écriture / de lecture dans le fichier de symboles, ou de ne pas les reprendre du tout.

Options pour le Base de données du projet

Dans cette boîte de dialogue, on définit si le projet doit être géré dans une base de données du projet et si tel est le cas, on procède aux configurations nécessaires de l'interface ENI.

Options de la boîte de dialogue de la catégorie Base de données du projet



Utiliser la base de données du projet (ENI): Activez cette option si vous souhaitez accéder à une base de données du projet par le biais d'un serveur ENI, de façon à traiter certains ou tous les modules se rapportant à ce projet via cette base de données. Il faut pour ce faire que le serveur ENI et la base de données soient installés et que vous soyez défini comme utilisateur dans chacun. Reportez vous à cet effet à la documentation du serveur ENI ainsi qu'au chapitre 7, 'L'interface CoDeSys-ENI'.

Si cette option est activée, les options de la base de données (check-in, édition) sont à disposition pour chaque objet du projet. D'une part, certaines fonctions de la base de données s'effectuent automatiquement, si cela a été configuré en conséquence dans les options de la boîte de dialogue, et d'autre part les commandes du menu 'Projet' 'Liaison avec la base de données' peuvent être utilisées pour un appel ciblé des fonctions. En outre, un onglet 'Liaison à la base de données' est disponible dans la boîte de dialogue pour les Propriétés, par lequel on peut attribuer une certaine catégorie de base de données à un module.

Déposer de nouveaux objets automatiquement dans la base de données du projet:

Vous devez procéder ici à un réglage standard : Si un objet est inséré pour la première fois dans le projet ('Insérer objet'), il est automatiquement classé sous la Catégorie d'objets ici configurée. Cette affectation est reproduite dans les caractéristiques des objets ('Projet' 'Objet' 'Propriétés') et peut également y être modifiée pour l'objet. Voici les affectations possibles:

Projet: Le module est créé dans le répertoire de la base de données définie dans la boîte de dialogue Configuration-ENI/Objets de projet sous 'Nom du projet'.

Objets communs: Le module est géré dans le répertoire de la base de données définie dans la boîte de dialogue Configuration-ENI/Objets communs sous 'Nom du projet'.

Local: Le module n'est pas géré via ENI dans la base de données, mais est uniquement enregistré localement dans le projet.

Outre 'Objets de Projet' et 'Objets communs', il existe une troisième Catégorie de base de données 'Fichiers de compilation' pour les objets créés seulement lors de la compilation d'un projet et pour lesquels cette catégorie n'est pas pertinente.

Demander le type de nouveaux objets: Si cette option est activée, la boîte de dialogue 'Projet' 'Objet' 'Propriétés' s'ouvre à chaque insertion d'un nouvel objet dans le projet; vous pouvez

sélectionner dans cette boîte de dialogue laquelle des trois catégories d'objets ci-dessus doit se rapporter au module. Vous pouvez ainsi écraser le réglage standard 'Déposer de nouveaux objets automatiquement dans ...'

Configurer ENI...: Ce bouton nous mène aux réglages ENI qui ont été subdivisés en trois boîtes de dialogue.

Les objets appartenant au projet et devant être gérés par la base de données peuvent être attribués aux catégories de bases de données 'Objets de projet', 'Objets communs' ou 'Fichiers de compilation'. Pour chacune de ces catégories, les boîtes de dialogue des options des bases de données du projet ci-dessous déterminent le répertoire dans lequel elles se trouvent et quelles valeurs par défaut valent pour certaines fonctions de bases de données :

- Dialogue Configuration ENI / Objets de Projet
- Dialogue Configuration ENI / Objets communs
- Dialogue Configuration ENI / Fichiers de compilation

Veuillez noter : les objets sont en tout cas également enregistrés localement - donc en même temps que le projet.

Les boîtes de dialogue apparaissent l'une à la suite de l'autre lors d'une première configuration, et un **Assistant** vous guide (via les boutons **Suivant/Précédent**) tout au long de ces étapes. Les paramètres introduits dans la première boîte de dialogue sont repris dans les deux autres et il ne faut alors introduire que les différences par rapport à ces paramètres.

Si une configuration est déjà disponible, les boîtes de dialogue sont reprises sous la forme de trois onglets dans la fenêtre.

Si, avant la configuration, il n'y a pas encore eu d'ouverture de session correcte à la base de données, (Boîte de dialogue d'ouverture de session via le menu 'Projet' 'Liaison avec la base de données' 'Login') une boîte de dialogue d'ouverture de session s'ouvre automatiquement à cet effet.

Options pour les Objets de Projet and les Objets communs

Ces boîtes de dialogue font partie intégrante de la configuration des options pour la base de données du projet ('Projet' 'Options' 'Base de données du projet). On y définit les paramètres d'accès des objets permettant la gestion des catégories 'Projet' et Objets communs' dans la base de données. Les deux boîtes de dialogue se composent des mêmes points. (Une troisième boîte de dialogue est disponible pour la configuration de la catégorie Fichiers de compilation).

Liaison ENI

Adresse TCP/IP:	Adresse de l'ordinateur sur lequel le serveur ENI tourne.
Port:	Valeur par défaut : 80; doit correspondre aux réglages de la configuration du serveur ENI.
Nom du projet:	Nom du répertoire de la base de données, sous lequel les objets de la catégorie concernée devraient être classés. Si le répertoire existe déjà dans la base de données, vous pouvez le sélectionner dans l'arborescence des projet ENI, que vous obtenez via le bouton .... Si toutefois vous n'êtes pas encore identifié comme utilisateur ENI par le biais de la Boîte de dialogue d'ouverture de session, cette même boîte de dialogue apparaît dès que vous appuyez sur ce bouton; vous devez alors introduire vos **Nom d'utilisateur** et **Mot de passe** permettant l'accès ENI aux trois catégories de bases de données.
Accès lecture seule	Si cette option est activée, vous ne pouvez accéder aux données des répertoires de base de données définis ici qu'en lecture seule.

Dialogue 'Objets de Projet' dans la catégorie 'Base de données du projet'

Objets de projet

Liaison ENI
Adresse TCP/IP: localhost
Port: 80
Nom du projet: mmm\project\
Accès lecture seule

Appeler
Ouvrir au projet — Avec demande
Tout de suite après des changements dans — Avec demande
Avant chaque compilation — Avec demande

Extraire
Tout de suite au début d'un changement — Avec demande

Archiver
Enregistrer sous projet — Avec demande
Apres compilation réussite — Avec demande

< Précédent | Suivant > | Annuler

Appeler

La fonction de base de données 'Appeler' (Menu 'Projet' 'Liaison avec la base de données' permet de copier la version en cours d'un module hors de la base de données sur un projet ouvert localement, en écrasant la version locale. Ceci se produit automatiquement, pour tous les modules modifiés par rapport à la version locale du projet activé et aux moments désirés (marqué d'un crochet) :

Ouvrir au projet	Lorsque le projet est ouvert dans CoDeSys
Tout de suite après des changements dans ENI	Au moment du check-in d'une version plus récente dans la base de données ; ce module est directement mis à jour dans le projet ouvert, et un message adéquat vous est alors donné.
Avant chaque compilation	Avant chaque compilation dans CoDeSys

Extraire

La fonction de base de données 'Extraire' permet de marquer le module comme étant 'en traitement' et de le bloquer à d'autres utilisateurs, et ce jusqu'à ce qu'il soit à nouveau libéré par le biais du check-in ou de l'annulation du check-out.

Si l'option **Tout de suite au début d'un changement** est activée, le check-out d'un module se produit automatiquement dès que son traitement à l'intérieur du projet a débuté. Si l'objet en question est inaccessible (check-out par un autre utilisateur, reconnaissable à la présence d'une croix rouge devant le nom de l'objet dans l'Organisateur d'objet), un message vous est délivré.

Archiver

La fonction de base de données 'Archiver' signifie qu'une nouvelle version de l'objet est créée dans la base de données. Les anciennes versions sont conservées. Voici les moments possibles:

Enregistrer sous projet	Si cette fonction est activée, un check-in de chaque module modifié se produit automatiquement à chaque sauvegarde du projet.
Apres compilation réussite	Si cette fonction est activée, un check-in de chaque objet modifié se produit après chaque compilation correcte du projet.

L'option **Avec demande** peut toujours être activée pour les points Appeler, Extraire et Archiver. Dans ce cas, un boîte de dialogue s'affiche avant que l'action concernée ne soit exécutée, incitant l'utilisateur à confirmer ou à interrompre cette action.

Les différents points de la boîte de dialogue '**Objets communs**' correspondent à ceux de la boîte de dialogue décrite plus haut 'Objets du Projet'. Les configurations valent pour tous les objets affectés à la catégorie 'Objets communs'.

Les boîtes de dialogue apparaissent l'une à la suite de l'autre lors d'une première configuration, et un **Assistant** vous guide (via les boutons **Suivant/Précédent**) tout au long de ces étapes. Les paramètres introduits dans la première boîte de dialogue sont repris dans les deux autres et il ne faut alors introduire que les différences par rapport à ces paramètres.

Si une configuration est déjà disponible, les boîtes de dialogue sont reprises sous la forme de trois onglets dans la fenêtre.

Options pour les Fichiers de compilation se rapportant à la base de données

Cette boîte de dialogue fait partie intégrante de la configuration des options pour la base de données du projet ('Projet' 'Options' 'Base de données du projet). On y définit la gestion des objets de la catégorie Fichiers de compilation dans la base de données. (En outre, deux autres boîtes de dialogue sont à disposition pour définir les options des objets de la catégorie Projet et de la catégorie Objets communs.)

Dialogue Fichiers de compilation in catégorie 'Base de données du projet

Fichiers de compilation

Liaison ENI
Adresse TCP/IP: localhost
Port: 80
Nom du projet: mmm\compile\ ...

☑ Créer une information de symboles ASCII
☐ Créer une information de symboles binaire
☐ Créer projet d'initialisation

< Précédent | Terminer | Annuler

Pour les champs **Adresse TCP/IP**, **Port**, **Nom du projet,** reportez-vous à la boîte de dialogue Objets du Projet/Objets communs.

Créer une information de symboles ASCII (*.sym)

Créer une information de symboles binaires (.sdb)

Si cette option est activée, tout fichier de symboles *.sym (format texte) ou *.sdb (format binaire) est aussi repris dans la base de données dès sa création. Lors de la création des symboles, les attributs d'objets configurés dans les options du projet sous la catégorie 'Configuration des symboles' restent valables.

Créer projet d'initialisation

Si cette option est activée, tout projet d'initialisation est aussi repris dans la base de données dès sa création.

Les boîtes de dialogue apparaissent l'une à la suite de l'autre lors d'une première configuration, et un **Assistant** vous guide (via les boutons **Suivant/Précédent**) tout au long de ces étapes. Les paramètres introduits dans la première boîte de dialogue sont repris dans les deux autres et il ne faut alors introduire que les différences par rapport à ces paramètres.

Si vous appuyez sur **Annuler**, vous revenez sur la boîte principale de dialogue, et les paramètres de l'onglet 'Fichiers de compilation' ne sont pas enregistrés. (Ceux qui avaient déjà été effectués sous Objets du Projet et Objets communs sont cependant conservés.)

Macros

Si vous sélectionnez cette catégorie dans la boîte de dialogue 'Options', vous obtenez les boîtes de dialogue suivantes :

Options de la boîte de dialogue de la catégorie Macros

Options
Catégorie:
Ouvrir & Fermer
Informations utilisateur
Editeur
Environnement de travail
Couleurs
Répertoires
Journal
Options de compilation
Mots de passe
Download Sourcecode
Configuration de symbole
Base de données du projet
Macros
Nom:
FileOpen
Macro:
Nouveau
Renommer
Bibliothèque des macros:
Inclure...
Créer...
OK
Annuler
Aide
Liste des macros:
Macros
Test.mac
Second. ma
FileOpen
Login
..\LibV2_2\first.mac
Compile
FileSave
Login
FileSaveTop
Menu:
FileOpen
Commandes:
file xxxx

Dans cette boîte de dialogue, vous pouvez définir des macros à partir du fichier des commandes de CoDeSys; ces macros peuvent alors être appelées dans le menu 'Editer' 'Macros'.

Veuillez procéder comme suit pour définir de nouvelles macros :

1. Introduisez dans le champ de saisie **Nom** le nom de la macro à créer. Après avoir appuyé sur le bouton **Nouveau**, ce nom est repris dans la **Liste des macros** et y est sélectionné. La liste des macros est organisée en une arborescence. Les macros créées localement apparaissent l'une sous l'autre, et les bibliothèques de macro éventuellement liées (voir ci-dessous) y apparaissent accompagnées du nom de fichier de bibliothèque. Vous pouvez ouvrir ou fermer la liste des éléments de la bibliothèque par le biais du signe plus ou moins avant le nom de la bibliothèque.
2. Définissez dans le champ **Menu** la dénomination de l'entrée de menu à laquelle la macro est rattachée dans le menu 'Editer' 'Macros'. Pour définir une lettre comme raccourci, il faut placer le symbole '&' avant cette lettre. *Exemple :* le nom "Ma&cro 1" produit l'entrée de menu "Ma<u>c</u>ro 1".

3. Introduisez maintenant dans le champ d'éditeur **Commandes**, les commandes se rapportant à la macro marquée dans la liste des macros. Toutes les commandes du mécanisme collectif et les mots-clés se rapportant à celles-ci sont autorisé(e)s, et vous pouvez en obtenir une liste par le biais du bouton **Aide**. Une nouvelle ligne d'instructions peut être insérée via <Ctrl><Entrée>. Vous obtenez le menu contextuel avec toutes les fonctions d'éditeur littéral par le biais du bouton droit de la souris. Les composantes de commande allant ensemble peuvent être regroupées à l'aide de guillemets.
4. Si vous souhaitez créer d'autres macros, répétez les étapes 1 à 3 avant de confirmer à l'aide du bouton OK et de fermer la boîte de dialogue.

Si vous souhaitez **effacer** une macro, sélectionnez-la dans la liste des macros et appuyez sur la touche <Suppr>.

Si vous souhaitez renommer une macro, sélectionnez-la dans la liste des macros, donnez-lui un autre nom sous **Nom** et appuyez sur le bouton **Renommer**.

Si vous souhaitez **modifier** une macro existante, sélectionnez-la dans la liste des macros et éditez les champs menu et/ou instructions. Les changement sont enregistrés avec OK.

En quittant la boîte de dialogue avec **OK**, la version actuelle de la macro est sauvegardée dans le projet.

Les entrées de menu de macros apparaissent dans l'ordre selon lequel elles ont été définies dans le menu 'Editer' 'Macros'. Une vérification de la macro ne se produit que lors de d'exécution de la commande du menu.

Bibliothèque des macros:

Les macros peuvent être enregistrées dans des bibliothèques externes de macros, et peuvent ainsi par exemple être liées à d'autres projets.

- Création d'un bibliothèque de macros constituée de macros du projet actuel : Appuyez sur le bouton **Créer...**. Vous obtenez la boîte de dialogue '**Copier des objets**' qui reprend toutes les macros disponibles. Marquez les macros souhaitées et confirmez avec OK. Suite à quoi la boîte de dialogue pour la sélection se referme et une nouvelle boîte de dialogue '**Enregistrer la bibliothèque des macros**' s'ouvre. Introduisez ici le nom et le chemin d'accès de la bibliothèque à créer et appuyez sur le bouton Enregistrer. La bibliothèque est ainsi créée sous **<nom de la bibliothèque >.mac**, et la boîte de dialogue se referme.
- Intégration de la bibliothèque de macros <nom de la bibliothèque>.mac dans le projet en cours d'utilisation.
  Appuyez sur le bouton **Inclure...**. La boîte de dialogue '**Ouvrir la bibliothèque des macros**' apparaît, affichant automatiquement les seuls fichiers avec l'extension *.mac. Sélectionnez la bibliothèque souhaitée et appuyez sur le bouton Ouvrir. La boîte de dialogue se referme et la bibliothèque apparaît sous forme d'arborescence reprenant la liste des macros.

Remarque : Les macros relatives à un projet peuvent également être exportées ('Projet' 'Exporter').

4.3 Gestion de projets

Les commandes relatives à un projet global se situent dans les menus 'Fichier' et 'Projet'. Voyez à cet effet les chapitre suivant.

'Fichier' - 'Nouveau'

Icône : 

Cette commande vous permet de créer un nouveau projet intitulé 'Sans nom'. Lors de l'enregistrement, il faudra modifier ce nom.

'Fichier' 'Nouveau de modèle'

Cette commande vous permet d'ouvrir un projet modèle. Un dialogue s'ouvre vous permettant de sélectionner un fichier de projet qui sera alors ouvert avec comme nom "Inconnu".

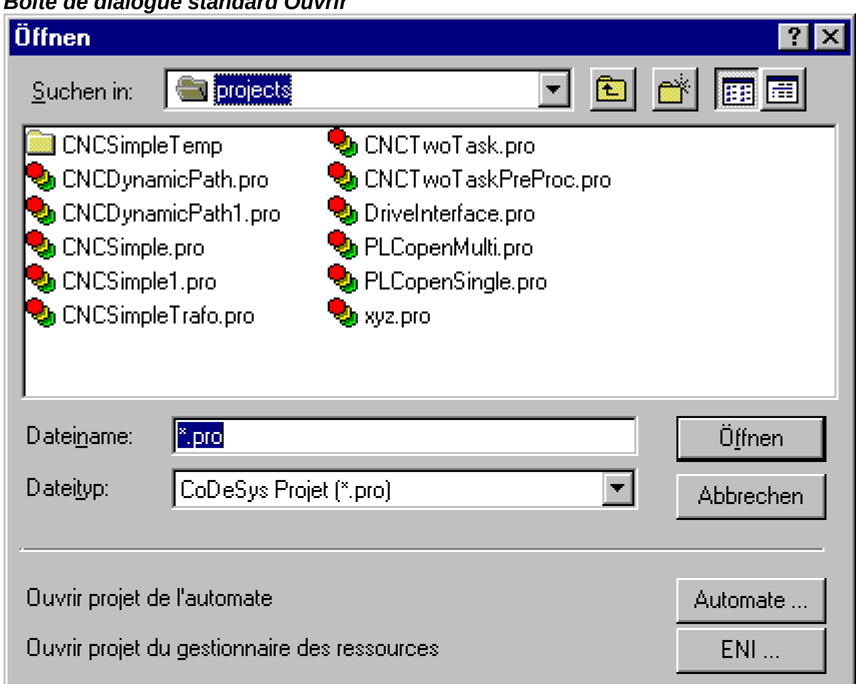
'Fichier' - 'Ouvrir'

Icône : 

Cette commande vous permet d'ouvrir un projet existant. Lorsqu'un projet est déjà ouvert et qu'il a été modifié, **CoDeSys** demande si vous souhaitez l'enregistrer ou non.

La boîte de dialogue Ouvrir s'affiche, dans laquelle vous devez sélectionner un fichier de projet portant l'extension "*.pro" ou un fichier bibliothèque portant l'extension "*.lib". Ces fichiers doivent exister, car la commande "**Ouvrir**" ne permet pas de créer un projet.

Boîte de dialogue standard Ouvrir



Ouvrir le projet de l'automate

Pour charger un projet d'un automate programmable, appuyez sur le bouton **PLC** dans **Ouvrir le projet de l'automate**. S'il n'existe pas encore de liaison avec l'automate programmable, vous accédez dans un premier temps à la boîte de dialogue Paramètres de communication (voir le menu 'En Ligne' 'Paramètres de communication') afin de configurer les paramètres de transmission. S'il existe déjà une liaison en ligne, le logiciel vérifie si des fichiers de projet portant le même nom figurent déjà dans le répertoire de votre ordinateur. Si c'est le cas, vous accédez à la boîte de dialogue **Charger le projet de l'automate** afin de déterminer si les fichiers locaux doivent être remplacés par les fichiers utilisés dans l'automate programmable. (Ce processus est l'inverse du processus 'En Ligne' 'Application téléchargée du code source dans l'automate', grâce auquel le fichier source du projet est enregistré dans l'automate. À ne pas confondre avec 'Créer projet d'initialisation' !)

Remarque : Nous attirons votre attention sur le fait qu'un projet d'automate programmable qui est chargé, ne possède pas encore de nom. Vous devez par conséquent enregistrer le projet sous un nouveau nom. Si le système cible le supporte, une 'Désignation' saisie au sein des informations sur le projet sera automatiquement attribuée comme nouveau nom de fichier. Dans ce cas, le dialogue de sauvegarde de fichier s'ouvrira automatiquement lors du chargement du projet à partir de l'API, ce dialogue reprenant ce nom de fichier et vous demandant de le confirmer ou de le modifier.

Dans le cas où aucun projet n'a encore été chargé dans l'automate programmable, le message d'erreur adéquat est affiché.

(Se reporter pour cela à la catégorie 'Sourcedownload' dans 'Projet' '**Options**')

Ouvrir projet du gestionnaire des ressources

Cette option sert à ouvrir un projet géré dans une base de données de projet ENI. Il faut pour ce faire avoir accès à un serveur ENI qui sert de base de données. Utilisez tout d'abord le bouton **ENI...** afin d'ouvrir la boîte de dialogue 'Objets de projet' permettant la connexion au serveur.

Introduisez les Données d'accès adéquates (Adresse TCP/IP, Port, Nom de l'utilisateur, Mot de passe, Accès lecture seule) et le répertoire de la base de données (Nom du projet) hors duquel les objets de la base de données doivent être appelés, et confirmez avec **Suivant**. Suite à quoi la boîte de dialogue se referme et la boîte de dialogue relative à la catégorie 'Objets communs' s'ouvre. Introduisez ici aussi les données d'accès. Vous refermez la boîte de dialogue et appelez automatiquement les objets du répertoire concerné avec **Terminer**. Vous pouvez maintenant procéder aux configurations souhaitées des options du projet, telles qu'elles vaudront pour le traitement ultérieur du projet. Si vous souhaitez continuer à gérer le projet dans le cadre de la base de données, configurez-le en conséquence dans les boîtes de dialogue de la catégorie Base de données du projet.

Les derniers projets utilisés

En dessous de la commande 'Fichier' '**Quitter**' sont affichés les derniers projets utilisés. Si vous en sélectionnez un, il s'ouvre.

Si l'accès au projet a été limité par des mots de passe ou des niveaux d'accès, une boîte de dialogue s'affiche, dans laquelle vous devez entrer le mot de passe pertinent.

'Fichier' 'Fermer'

Cette commande vous permet de fermer le projet actuellement ouvert. Si le projet a été modifié, **CoDeSys** demande si les modifications apportées doivent être mémorisées ou non.

Lorsque le projet à enregistrer est intitulé "Sans nom", il convient de lui donner un nom (voir 'Fichier' 'Enregistrer sous').

'Fichier' 'Enregistrer'

Icône : **Raccourci : <Ctrl>+<S>**

Cette commande permet d'enregistrer tout projet qui a subi des modifications.

Lorsque le projet à enregistrer est intitulé "Sans nom", il convient de lui donner un nom (voir 'Fichier' 'Enregistrer sous').

'Fichier' 'Enregistrer sous'

Grâce à cette commande, vous pouvez stocker le projet actuel dans un autre fichier ou sous la forme d'une bibliothèque. Cette opération n'a pas d'incidence sur le fichier de projet d'origine.

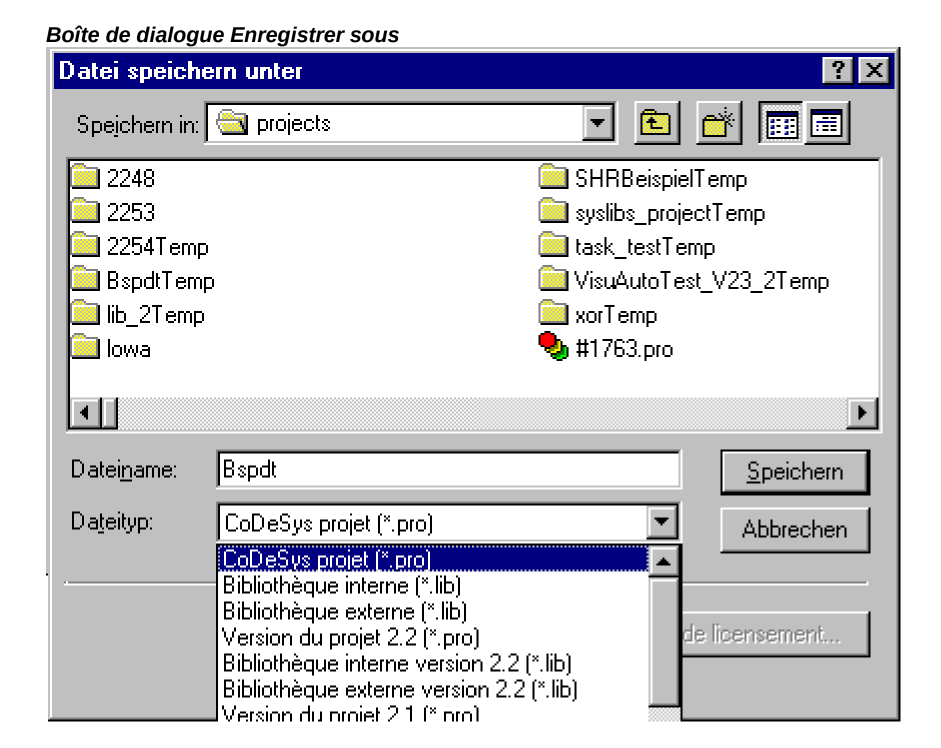
Lorsque vous avez sélectionné la commande, vous accédez à la boîte de dialogue pour l'enregistrement. Choisissez soit un **Nom de fichier** existant, soit entrez un nouveau nom et sélectionnez le **Type de fichier** souhaité.

Dans le second cas, choisissez le type de fichier **CoDeSys Projet (*.pro)**.

En sélectionnant le type de fichier **Projet Version 1.5 (*.pro)** ou **Projet Version 2.0 (*.pro)** ou **Projet Version 2.1 (*.pro),** ou **Projet Version 2.2 (*.pro)** le projet actuel est enregistré comme s'il avait été créé en version 1.5, 2.0, 2.1 ou 2.2**)** Attention, les données spécifiques à la version 2.1 peuvent être perdues! Mais le projet peut continuer à être géré en version 1.5, 2.0, 2.1 ou 2.2.

Vous avez également la possibilité d'enregistrer le projet actuel sous forme de bibliothèque, de façon à pouvoir l'utiliser dans d'autres projets. Choisissez le type de fichier **Librairie interne (*.lib)**, si vous avez programmé vos modules dans **CoDeSys**.

Boîte de dialogue Enregistrer sous



Choisissez le type de fichier **Librairie externe (*.lib)**, lorsque vous voulez intégrer des modules qui ont été implémentés dans d'autres langages de programmation (p. ex. le langage C). Cela signifie qu'un fichier supplémentaire sera enregistré sous le nom de fichier de la bibliothèque mais avec l'extension "*.h". Ce fichier est structuré comme un fichier C-Header et comporte les déclarations de tous les modules, types de données et variables globales.

Protection d'une bibliothèque par l'attribution d'une licence :

Si la bibliothèque doit être soumise à une licence, les informations nécessaires relatives à la licence peuvent lui être adjointes. À cet effet, il faut ouvrir la boîte de dialogue 'Editer les informations relatives à l'attribution d'une licence' par le biais du bouton **Edit info de licence**. Reportez-vous à cet effet à la description de la Gestion de licences .

Pour terminer, cliquez sur **OK**. Le projet actuel est enregistré dans le fichier spécifié. Si le nouveau nom donné au fichier existe déjà, un message vous demande si vous souhaitez remplacer le fichier concerné.

Lorsque vous enregistrez le fichier sous la forme d'une bibliothèque, la totalité du projet est compilée. En cas d'erreur, un message vous indique que, pour être enregistré sous la forme d'une bibliothèque, un projet ne doit comporter aucune erreur. Le projet n'est donc pas enregistré sous la forme d'une bibliothèque.

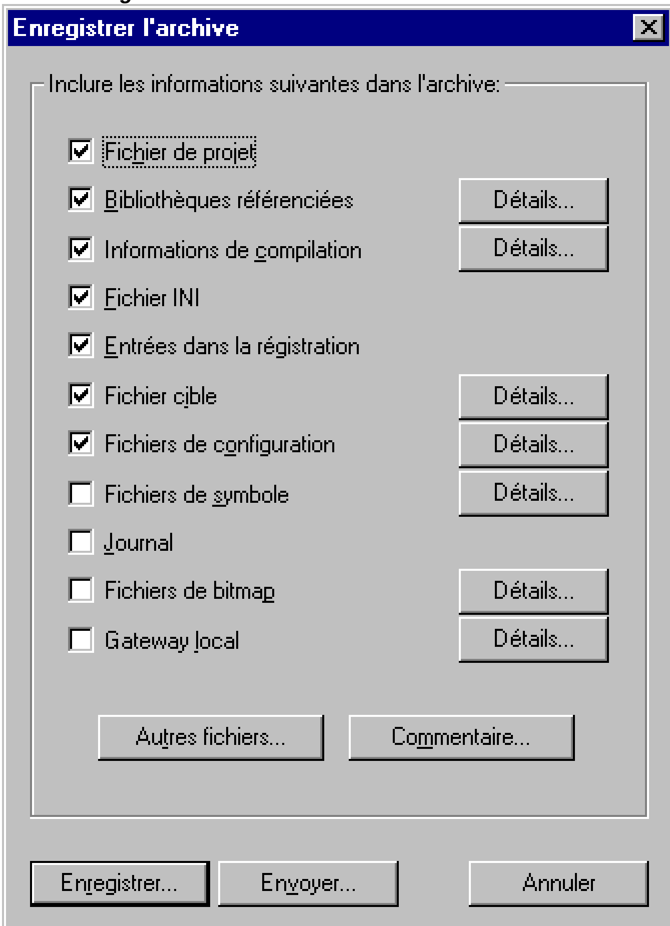
'Fichier' 'Enregistrer archive/envoyer'

Vous pouvez créer un fichier zip comprenant tous les fichiers relatifs à un projet **CoDeSys** à l'aide de cette commande. Ce fichier zip peut être enregistré dans le système de fichiers ou être envoyé directement par e-mail.

Veuillez noter: Cette fonction d'archivage ne convient pas pour rétablir des environnements de projets. Elle sert tout simplement à résumer tous les fichiers appartenant au projet. Lorsqu'une archive est décomprimée, il faut adapter le chemin de chaque fichier à l'environnement CoDeSys correspondant !

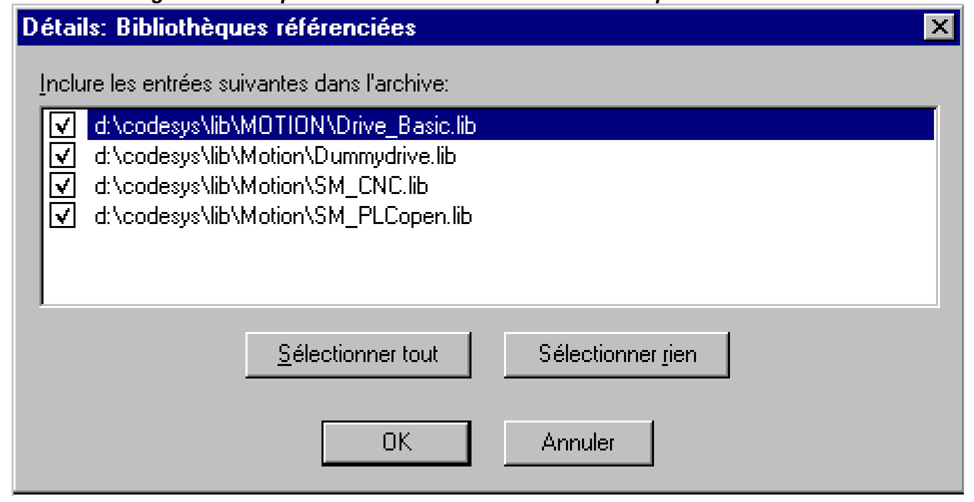
Après avoir exécuté cette commande, la boîte de dialogue 'Enregistrer le fichier d'archives' s'ouvre.

Écran 'Enregistrer le fichier d'archives'



Les catégories de fichiers devant être ajoutées au fichier d'archives du projet sont définies ici. Une catégorie est sélectionnée si la case en regard de cette catégorie est munie d'un crochet ☑. Pour ce faire, il vous suffit de cliquer une fois dans la case ou de cliquer deux fois sur la désignation de la catégorie. Pour chaque catégorie sélectionnée, tous les fichiers pertinents sont en principe copiés dans le fichier zip (voir tableau ci-dessous). Pour certaines catégories cependant, vous pouvez procéder à une sélection partielle. À cet effet, vous disposez de la boîte de dialogue 'Détails' que vous pouvez ouvrir par le biais du bouton **Détails** en regard.

Boîte de dialogue 'Détails' pour une sélection ciblée de fichiers pour le fichier d'archives



La boîte de dialogue 'Détails' vous donne une liste de tous les fichiers disponibles dans cette catégorie. Activez ou désactivez les fichiers souhaités : Les boutons **Sélectionner tout** et **Ne rien sélectionner** agissent sur tous les fichiers de la liste, un clic dans une case (dés)active le fichier en

regard, de même qu'un double-clic sur une entrée. En outre, si une entrée est marquée, vous pouvez l'activer ou la désactiver au moyen de la touche <Entrée>.

Lorsque vous fermez la boîte de dialogue avec OK, la sélection effectuée est acceptée. La configuration restera en mémoire jusqu'à la création définitive du fichier d'archives.

Dans la boîte principale de dialogue 'Enregistrer les archives', on reconnaît les catégories pour lesquelles une sélection partielle a été effectuée à l'arrière plan gris de la case : ..

Le tableau suivant vous indique quelles catégories de fichiers sont prédéfinies et quels fichiers s'y rapportent :

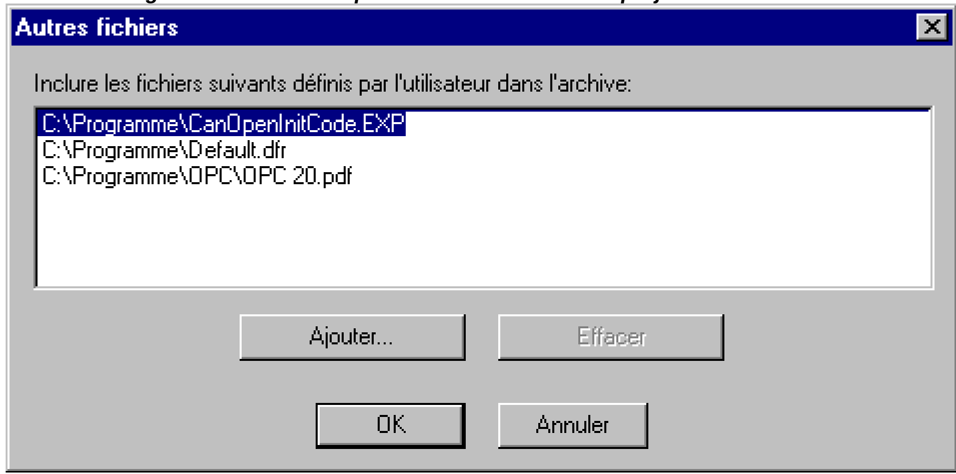
Catégorie	**Fichiers s'y rapportant**
Fichier de projet	<Nom du projet>.pro (le fichier de projet **CoDeSys**)
Bibliothèques référencées	*.lib, *.obj, *.hex (les bibliothèques et, le cas échéant, les fichiers .obj et .hex)
Informations de compilation	*.ci (Informations au sujet de la dernière compilation), *.ri (Information au sujet du dernier téléchargement) <temp>.* (compilation temporaire et fichiers de téléchargement) également pour la simulation
Fichier INI	Codesys.ini

Entrées dans la registry	Registry.reg (Entrées pour Automation Alliance, gateway et API; entrées suivantes de la base de registres :
	`HKEY_LOCAL_MACHINE\SOFTWARE\3S-Smart Software Solutions`
	`HKEY_LOCAL_MACHINE\SOFTWARE\AutomationAlliance"`

Fichier cible	*.trg (Fichiers cibles en format binaire pour toutes les cibles installées) *.txt (Fichiers cibles en format texte pour toutes les cibles installées, si disponibles)
Fichiers de configuration	Fichiers pour la configuration de l'automate (fichiers de configuration, fichiers d'outils, icônes, etc.) : p.ex. *.cfg, *.con, *.eds, *.dib, *.ico ....
Fichiers de symbole	*.sdb, *.sym (informations relatives aux symboles créées à partir du projet)
Journal	*.log (Journal du projet)
Fichiers de bitmap	*.bmp (Images bitmap utilisées dans les modules du projet et la visualisation)
Gateway local	Fichiers de gateway : Gateway.exe, GatewayDDE.exe, GClient.dll, GDrvBase.dll, GDrvStd.dll, Ghandle.dll, GSymbol.dll, GUtil.dll, et le cas échéant, d'autres DLL disponibles dans le répertoire de la gateway.

Pour ajouter d'autres fichiers à votre guise dans le fichier d'archives, ouvrez la boîte de dialogue **Autres fichiers...** via le bouton du même nom.

Boîte de dialogue 'Autres fichiers' pour fichier d'archives du projet



Vous pouvez créer ici votre propre liste de fichiers. Ouvrez à cet effet au moyen du bouton **Ajouter...** la boîte de dialogue standard permettant d'ouvrir un fichier. Choisissez un fichier et confirmez avec **Ouvrir**. Ce fichier est alors inclus dans la liste de la boîte de dialogue 'Autres fichiers'. Vous pouvez effacer une entrée de la liste au moyen du bouton Effacer. Lorsque la liste est terminée, fermez la boîte de dialogue avec OK, et les entrées seront sauvegardées jusqu'à la création du fichier zip.

Pour ajouter un fichier Lisezmoi au fichier zip, appuyez sur le bouton **Commentaire...**. Une boîte de dialogue du même nom s'ouvre, contenant un champ d'édition. Vous pouvez introduire ici du texte à votre guise. Lorsque vous refermez la boîte de dialogue avec OK, un fichier **Lisezmoi.txt** est crée en même temps que le fichier zip. Ce fichier Lisezmoi contient le texte que l'utilisateur a saisi ainsi que les ajouts automatiques de date de création et numéro de la version CoDeSys utilisée.

Générer le fichier zip :

Lorsque vous avez procédé à toutes les configurations souhaitées, le fichier zip peut être créé par le biais de la boîte principale de dialogue. Les boutons suivants sont disponibles:

- Enregistrer... crée et sauvegarde le fichier zip. Le dialogue standard pour la sauvegarde d'un dossier s'ouvre et vous pouvez indiquer l'endroit où ce fichier doit être classé. Par défaut, ce fichier zip porte le nom <Nom du projet>.zip. Si vous appuyez sur Enregistrer, la création du fichier d'archives débute. Le déroulement est accompagné d'un dialogue d'avancement et est consigné dans la fenêtre de messages.
- Envoyer… crée un fichier zip temporaire et génère automatiquement un e-mail vide qui contient le fichier zip en annexe. Cette fonction implique une installation correcte du MAPI (Messaging Application Programming Interface). Alors que l'e-mail est créé, un dialogue d'avancement apparaît et le déroulement est consigné dans la fenêtre de messages. Le fichier zip temporaire est effacé dès qu'il est repris comme annexe de l'e-mail.
- Annuler :La boîte de dialogue est fermée sans qu'il y ait création de fichier zip, et les configurations effectuées ne sont pas enregistrées.

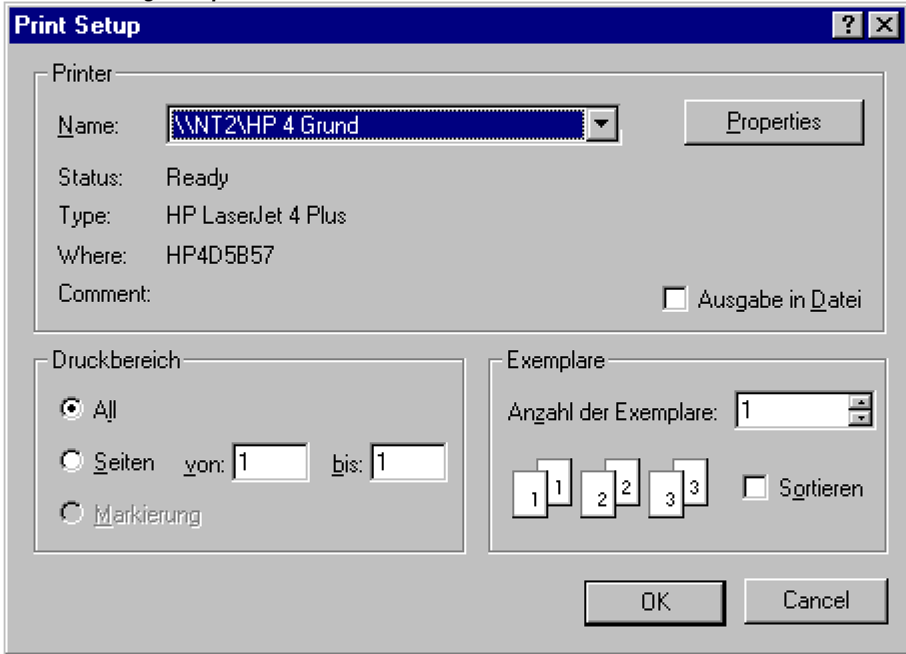
'Fichier' 'Imprimer'

Raccourci : <Ctrl>+<P>

Cette commande vous permet d'imprimer le contenu de la fenêtre active.

Après avoir sélectionné la commande, vous accédez à la boîte de dialogue Imprimer. Choisissez l'option souhaitée ou configurez l'imprimante, puis cliquez sur **OK**. Le contenu de la fenêtre active s'imprime. Des impressions en couleurs sont possibles à partir de tous les éditeurs.

Boîte de dialogue Imprimer



Dans la boîte de dialogue Imprimer, vous pouvez choisir la zone d'impression au niveau de Documentation du projet (soit **Tout** soit une sélection de **Pages** explicitement indiquées); pour ce qui est des objets, ils sont imprimés dans leur totalité. Vous pouvez spécifier le **Nombre de copies** et imprimer dans un fichier.

Le bouton **Propriétés** permet d'ouvrir la boîte de dialogue Configuration de l'imprimante.

Vous pouvez déterminer la mise en page du document que vous souhaitez imprimer via la commande 'Configuration Documentation' du menu 'Fichier'.

Pendant l'impression, une boîte de dialogue vous indique le nombre de pages déjà imprimées. Si vous fermez cette boîte de dialogue, l'impression cesse à la page suivante.

Pour documenter l'ensemble de votre projet, utilisez la commande 'Projet' 'Documentation du projet'.

Si vous souhaitez créer un document modèle pour votre projet, dans lequel vous souhaitez introduire tous les commentaires relatifs aux variables utilisées dans ce projet, ouvrez une liste globale de variables et utilisez la commande 'Extras' 'Créer document modèle'.

'Fichier' 'Configuration Documentation'

Cette commande vous permet de configurer les pages imprimées et donne accès à la boîte de dialogue affichée ci-dessous:

Boîte de dialogue Configurations de documentation

Configurations de Documentation
Cadre
Fichier: DEFAULT.DFR
Balayer...
Exploiter...
Variable: {Page} {FileName} {Date} {POUName} {Content}
Nouvelle page pour chaque
Nouvelle page pour chaque
OK
Configuration
Annuler

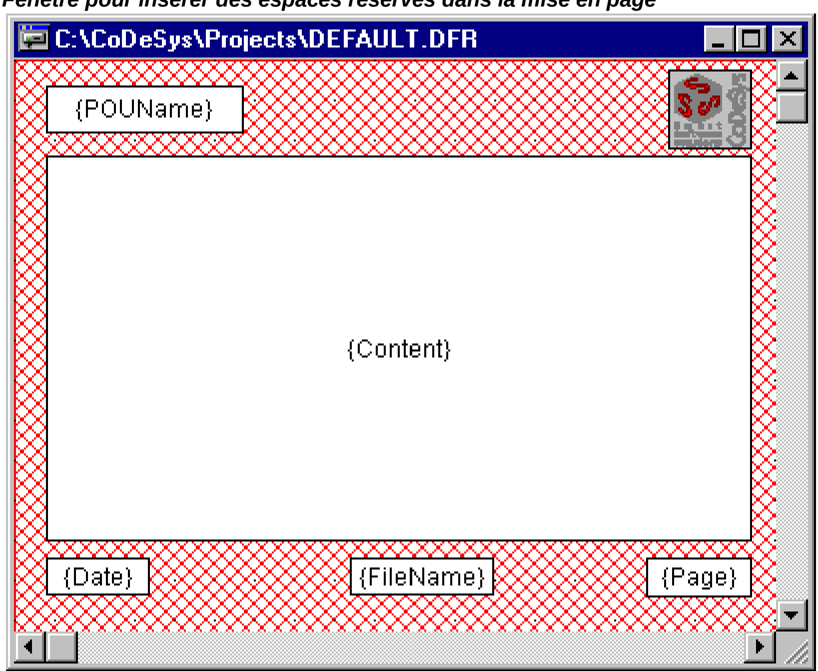
Vous pouvez entrer le nom du fichier dans lequel la configuration doit être mémorisée au niveau de la zone **Fichier** et lui donner l'extension ".dfr". Par défaut, le modèle est enregistré dans le fichier DEFAULT.DFR.

Si vous souhaitez modifier une configuration existante, cliquez sur le bouton **Parcourir** pour rechercher le fichier souhaité dans l'arborescence de répertoires.

De même, vous pouvez choisir de faire commencer une **nouvelle page pour chaque objet** et une **nouvelle page pour chaque sous-objet**. Le bouton **Configuration** vous permet d'accéder à la configuration de l'imprimante.

En cliquant sur le bouton **Modifier**, vous accédez au modèle de configuration de la mise en page. Dans ce fichier, vous pouvez indiquer les numéros de page, la date, le nom du fichier et du module ainsi que placer des graphiques et spécifier les zones dans lesquelles la documentation doit être imprimée. La surface de feuille prise en compte par la configuration de l'imprimante est hachurée en rouge.

Fenêtre pour insérer des espaces réservés dans la mise en page



L'option '**Espace réservé**' du menu 'Insérer' vous permet d'insérer un espace réservé. Vous avez le choix entre les cinq espaces réservés suivants (**Page, Nom de module, Nom de fichier, Date, Contenu**) et vous pouvez les placer sur la page en dessinant un rectangle. Ces espaces réservés ont les effets suivants:

Commande	**Espace réservé**	**Effet**
Page	{Page}	Impression du numéro de la page actuelle.
Nom de module	{POUName}	Impression du nom du module actuel.
Nom de fichier	{FileName}	Impression du nom du projet.
Date	{Date}	Impression de la date du jour.
Contenu	{Content}	Impression du contenu du module.

En outre, vous pouvez insérer une image Bitmap à l'aide de l'option '**Bitmap**' du menu 'Insérer' (p. ex. le logo d'une société). Pour cela, vous devez choisir le graphique à insérer, puis tracer un nouveau rectangle sur la page à l'aide de la souris. Il est possible d'insérer d'autres éléments de visualisation (voir le manuel 'CoDeSys Visualisation').

Lorsque le modèle a été modifié, **CoDeSys** demande, lors de la fermeture de la fenêtre, si vous souhaitez conserver ces modifications ou non.

'Fichier' 'Quitter'

Raccourci : <Alt>+<F4>

Cette commande permet de quitter **CoDeSys**.

Lorsqu'un projet est ouvert et que vous choisissez cette commande, le projet est fermé conformément à la description faite dans la partie 'Fichier' - 'Enregistrer'.

'Projet' 'Compiler'

Raccourci : <F11>

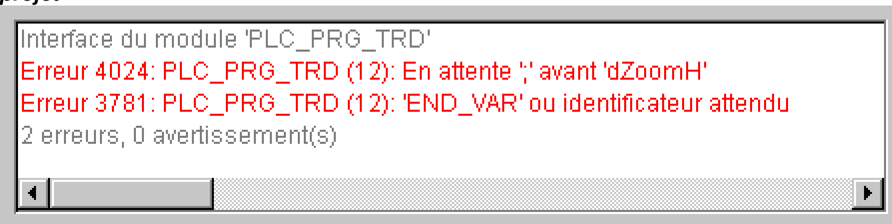
Vous compilez le projet à l'aide de 'Projet' 'Compiler'. La compilation est fondamentalement incrémentale, c.-à-d. les modules modifiés sont seuls compilés. Une compilation non incrémentale peut également être réalisée à partir de cette commande si vous avez auparavant exécuté la commande 'Projet' 'Réorganiser tout'.

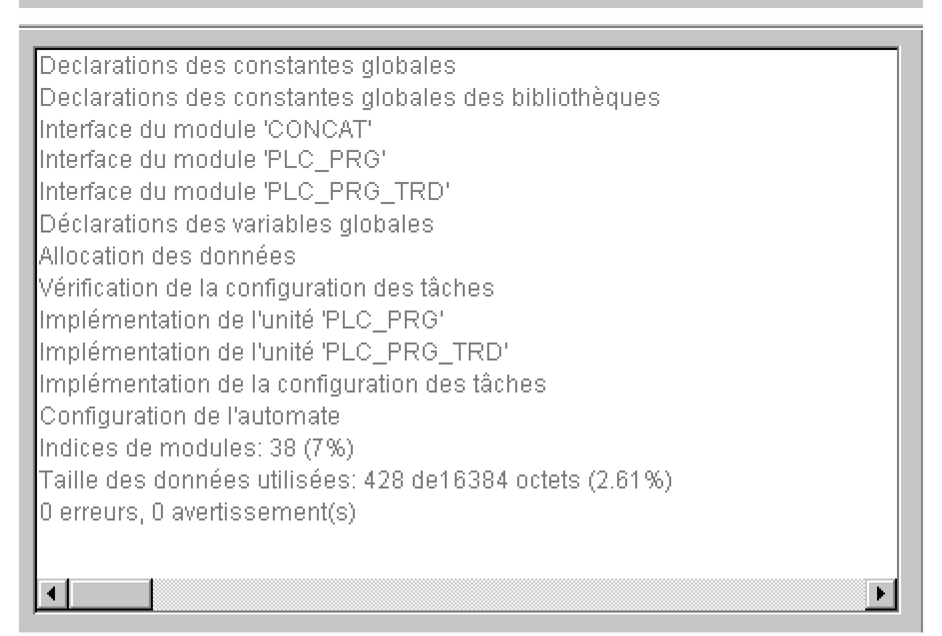
Pour les systèmes cible supportant les changements En ligne, tous les modules du gestionnaire des objets devant être chargés sur l'automate lors du prochain téléchargement sont marqués après la compilation d'une flèche bleue.

La compilation effectuée par le biais de 'Projet' 'Compiler' s'effectue automatiquement si vous êtes inscrit auprès de l'automate via 'En Ligne' 'Accéder au système'.

La fenêtre de messages s'ouvre lors de la compilation, indiquant l'avancement de la compilation, les erreurs éventuellement apparues pendant l'opération, les avertissements, ainsi que les données relatives aux indices ou à l'espace mémoire utilisé (avec un nombre et un pourcentage). Les erreurs et les avertissements sont indiqués par des numéros. Par le biais de F1, vous pouvez obtenir plus d'informations sur l'erreur marquée.

Exemple de messages d'erreur et d'informations de compilation pouvant apparaître dans la fenêtre de messages d'un projet





Toutes les erreurs possibles sont répertoriées dans l'annexe.

En activant l'option **Enregistrer avant la compilation** dans la boîte de dialogue Options, catégorie Charger & Enregistrer, le projet est enregistré avant la compilation.

Remarque : Les références croisées sont créées pendant la compilation et sont mémorisées dans les informations de compilation. Pour pouvoir utiliser les commandes **'Afficher diagramme d'appel', 'Afficher liste de références croisées'** et les commandes 'Variables non-utilisées', 'Accès concurrent' et 'Ecriture multiple sur la sortie' du menu 'Projet' **'**Enregistrer modifications**'** , le projet doit être à nouveau compilé après une modification.

'Projet' 'Compiler tout'

À l'inverse de la compilation incrémentale ('Projet' 'Compiler'), la commande 'Projet' 'Compiler tout' permet de compiler le projet dans son entièreté. Cependant, toutes les informations relatives au téléchargement ne sont pas rejetées, comme c'est le cas pour la commande 'Réorganiser tout'.

'Projet' 'Réorganiser tout'

À l'aide de cette commande, toutes les informations relatives au dernier téléchargement et à la dernière compilation sont effacées.

Après avoir sélectionné cette commande, une boîte de dialogue s'ouvre vous indiquant qu'un accès au système ne sera pas possible sans nouveau téléchargement. Vous pouvez ici confirmer ou interrompre cette commande.

Remarque : Une ouverture de session après 'Réorganiser tout' n'est possible que si le fichier *.ri contenant les informations du projet relatives au dernier téléchargement a auparavant été renommé ou copié en dehors du répertoire du projet (voir 'Charger les de download') et si ce même fichier peut à nouveau être rechargé explicitement avant l'ouverture de session.

'Projet' 'Charger les informations de download'

À l'aide de cette commande, vous pouvez à nouveau charger de manière ciblée les informations de téléchargement à partir d'un fichier *.ri. . . La boîte de dialogue standard 'Ouvrir fichier' s'ouvre lors de cette commande.

À chaque téléchargement, et éventuellement (selon le système cible) lors de la création de tout projet d'initialisation en mode Hors ligne, les informations relatives au téléchargement sont automatiquement sauvegardées dans un fichier dénommé **<Nom du projet><Identificateur cible>.ri** et classé dans le répertoire du projet. Ces informations sont automatiquement chargées à chaque ouverture du projet et servent, d'une part, lors d'une nouvelle ouverture de session, à vérifier si le projet correspond au projet qui vient d'être ouvert sur l'automate (vérification d'identité) ; d'autre part, on vérifie pour quels modules le code généré a été modifié. Seuls ces modules seront à nouveau chargés lors d'un téléchargement dans les systèmes supportant les changements en ligne. Ce fichier *.ri est ainsi une condition préalable à un changement En ligne.

Veuillez dès lors noter: La commande 'Projet' 'Réorganiser tout' efface automatiquement du répertoire de projet le fichier *.ri appartenant à ce même projet, si bien qu'aucun changement En ligne ne soit ensuite possible, à moins que le fichier *.ri n'ait été enregistré en un autre emplacement ou sous un autre nom et qu'il puisse à nouveau être chargé de manière ciblée.

'Projet' 'Traduire dans d'autres langues'

Cette option sert à traduire ou visualiser le fichier de projet en cours d'utilisation dans une autre langue. Ceci se produit par la lecture d'un fichier de traduction créé à partir du projet et complété à l'aide d'un éditeur littéral avec des textes traduits dans la langue souhaitée.

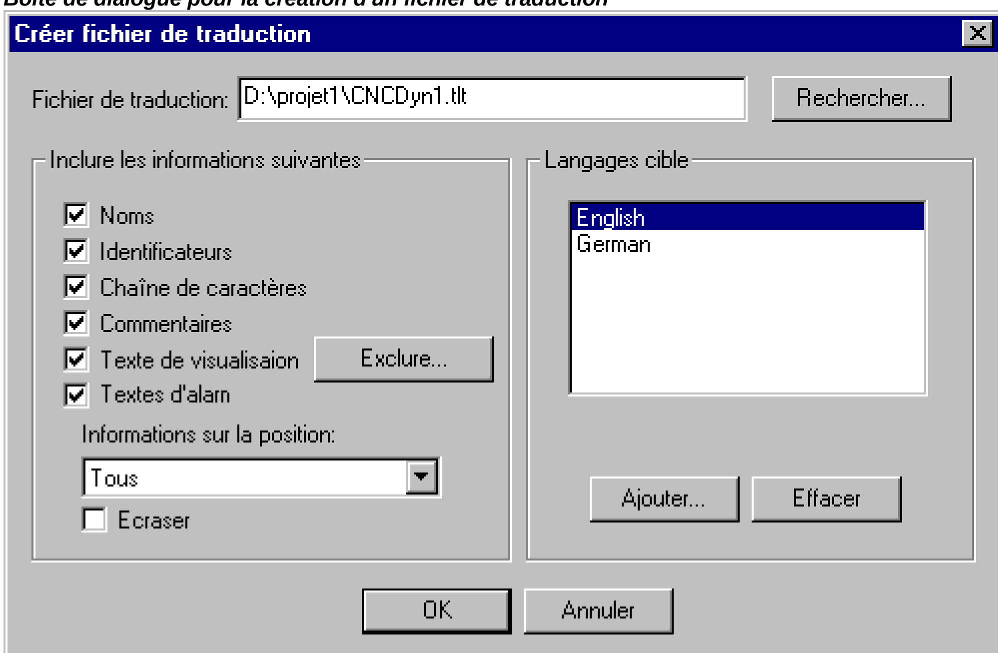
Vous disposez pour ceci de sous-menus :

- Créer un fichier de traduction...
- Traduire projet...
- Montrer le projet traduit
- Commuter traduction

'Projet' 'Créer fichier de traduction

Cette commande du menu 'Projet' 'Traduire dans une autre langue' active une boîte de dialogue 'Créer un fichier de traduction'.

Boîte de dialogue pour la création d'un fichier de traduction



Introduisez dans le champ **Fichier de traduction:** un chemin d'accès indiquant l'endroit où le fichier doit être sauvegardé. L'extension du fichier est par défaut *.tlt, il s'agit un fichier texte. L'extension ***.txt** est également possible, et même recommandée si par exemple le fichier doit être traité par le biais de WORD ou EXCEL, vu que dans ces cas, les données sont disposées sous forme de tableau.

S'il y a déjà un fichier de traduction que vous souhaitez traiter, introduisez le chemin d'accès de ce fichier ou utilisez la boîte de dialogue standard Windows permettant de sélectionner un fichier par le biais du bouton **Rechercher**.

Les informations suivantes provenant du projet peuvent être données en option en plus du fichier de traduction à remplacer ou à modifier, de telle sorte que vous disposiez, dans ce fichier de traduction, de : **Noms** (p.ex. le titre 'Module' dans l'organisateur d'objets), **Identificateurs**, **Chaîne de caractères, Commentaires**, **Textes de visualisation**, **Textes d'alarme**. Des **Informations sur la position** par rapport à ces éléments de projets peuvent également être reprises.

Si les options correspondantes sont pourvues d'un crochet, les informations sont reprises comme symboles linguistiques du projet en cours d'utilisation dans un nouveau fichier de traduction à créer, ou ajoutées à un fichier de traduction existant. Si l'option n'est pas sélectionnée, toutes les informations relatives à la catégorie concernée sont effacées du fichier de traduction, quel que soit le projet d'où elles proviennent.

Les textes de visualisation utilisés ici sont les éléments de visualisation 'Texte' et 'Texte pour info-bulle'.

Remarque : À propos des textes de visualisation ('Texte' et 'Texte pour info-bulle'), il faut veiller à ce qu'ils soient saisis dans la boîte de dialogue des éléments de visualisation entourés de symboles "#" (p.ex. #texte#) de façon à ce qu'ils puissent être repris dans le fichier de traduction. (Voir à cet effet le manuel 'CoDeSys Visualisation') Ces textes ne seront pas traduits par le biais de la commande 'Projet' 'Traduire dans une autre langue'. Un changement de langue pour la visualisation ne peut s'effectuer qu'en mode En ligne, en sélectionnant la langue correspondante dans la boîte de dialogue 'Extras' 'Configurations'.

Informations sur la position : À l'aide des données Chemin d'accès au fichier, Module, Ligne, celles-ci décrivent la position du symbole linguistique mis à disposition pour la traduction. Vous avez ici le choix entre trois options:

- Pas d'information: il n'y a pas d'informations de position générées.

- Première occurrence: La position de la première occurrence de l'élément à traduire est reprise dans le fichier de traduction.
- Tous: Toutes les positions auxquelles l'élément concerné intervient dans le projet vous sont indiquées.

Si un fichier de traduction créé auparavant est édité et comporte plus d'informations de position que ceux mis à disposition par la sélection, celles-ci sont abrégées en conséquence ou entièrement effacées, quel que soit le projet d'où elles ont été générées.

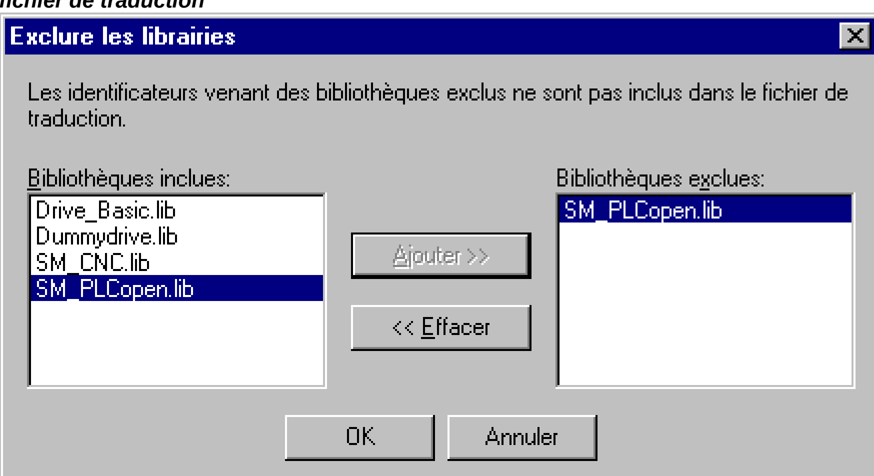
Remarque : Par élément (symbole linguistique), on ne peut générer qu'un maximum de 64 informations sur la position, même si l'utilisateur a sélectionné 'Tous' sous la rubrique "Informations sur la position" dans la boîte de dialogue du fichier de traduction.

Ecraser: Toutes les informations existantes sur la position dans le fichier de traduction en cours de traitement sont écrasées, quel que soit le projet d'où elles ont été générées.

Langages cible: Cette liste contient des désignations pour toutes les langues comprises dans le fichier de traduction ou qui doivent être reprises après avoir fermé la boîte de dialogue 'Créer fichier de traduction'.

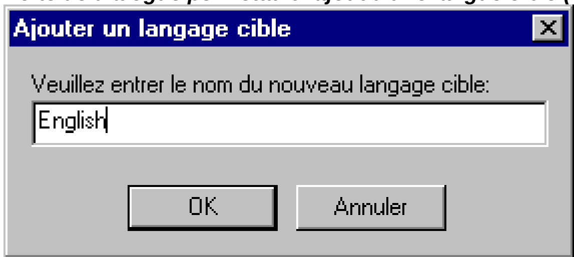
Le bouton **Exclure** ouvre la boîte de dialogue 'Exclure les librairies'. Vous pouvez sélectionner ici, dans la liste des bibliothèques associées au projet, toutes celles dont les identificateurs ne sont pas repris dans le fichier de traduction. L'entrée concernée est sélectionnée à l'aide de la souris dans le tableau de gauche **Bibliothèques inclues** et ajoutée à la liste dans le tableau de droite **Bibliothèques exclues** à l'aide du bouton **Ajouter**. De la même manière, le bouton **Effacer** permet d'effacer une entrée sélectionnée à droite. La configuration est confirmée avec **OK** et la boîte de dialogue se referme.

Boîte de dialogue permettant d'exclure des informations venant de certaines bibliothèques dans l'élaboration du fichier de traduction



Le bouton **Ajouter** ouvre la boîte de dialogue **'Ajouter une langue cible'**:

Boîte de dialogue permettant l'ajout d'une langue cible (Projet, Traduire dans une autre langue)



Un identificateur de langue doit être saisi dans le champ d'édition, et celui-ci ne peut contenir un espace au début ou à la fin ou encore un tréma (ä, ö, ü).

Vous refermez la boîte de dialogue avec **OK**, et la nouvelle langue cible apparaît dans la liste des langues cibles.

Le bouton **Effacer** permet d'effacer l'entrée sélectionnée de la liste.

Vous pouvez également confirmer la boîte de dialogue 'Créer un fichier de traduction' à l'aide du bouton **OK** de manière à générer un fichier de traduction. S'il y a déjà un fichier de traduction du même nom, une demande de sécurité apparaît tout d'abord, vous incitant à répondre par oui ou par non :

```
"Le fichier de traduction donné existe déjà. Il va être modifié en conséquence, et
une copie de sécurité du fichier existant déjà sera réalisée. Voulez-vous
continuer ?"
```

Non permet de revenir à la boîte de dialogue 'Créer un fichier de traduction' sans modification. Si **Oui** a été choisi, une copie du fichier de traduction existant déjà est réalisée, portant le nom "Backup_of_<Fichier de traduction>.xlt" et classée dans le même répertoire, et le fichier de traduction concerné est modifié conformément aux options sélectionnées.

Lors de la création d'un fichier de traduction :

- Pour chaque nouvelle langue cible, un espace réservé („##TODO") est généré pour chaque symbole linguistique affiché.
- Si vous traitez un fichier de traduction déjà existant, les entrées dans le fichier relatives aux langues présentes dans le fichier de traduction mais non reprises dans la liste des langues-cibles sont effacées, quel que soit le projet d'où elles ont été générées.

Edition du fichier de traduction

Le fichier de traduction est à ouvrir et sauvegarder sous forme de fichier texte. Les symboles ## identifient des mots-clés. Les espaces réservés ##TODO dans le fichier peuvent être remplacés par les textes de traduction adéquats. Un extrait délimité par ##NAME_ITEM et ##END_NAME_ITEM est créé pour chaque symbole linguistique (Pour les commentaires, ##COMMENT_ITEM et ainsi de suite).

Reportez-vous à l'exemple ci-dessous d'un extrait du fichier de traduction relatif au nom d'un module utilisé dans le projet : ST_Visualisierung. Les langues cibles anglais (USA) et français sont prévues. Dans cet exemple, les informations sur la position pour l'élément de projet à traduire sont inclus :

avant la traduction :

```
##NAME_ITEM
[D:\CoDeSys\projects\Bspdt_22.pro::ST_Visualisierung::0]
ST_Visualisierung
##English :: ##TODO
##French :: ##TODO
##END_NAME_ITEM
```

après la traduction :

les traductions en anglais et en français de 'Visualisierung' ont été introduites en lieu et place de #TODO :

```
##NAME_ITEM
[D:\CoDeSys\projects\Bspdt_22.pro::ST_Visualisierung::0]
ST_Visualisierung
##English :: ST_Visualization
##French :: ST_Visu
##END_NAME_ITEM
```

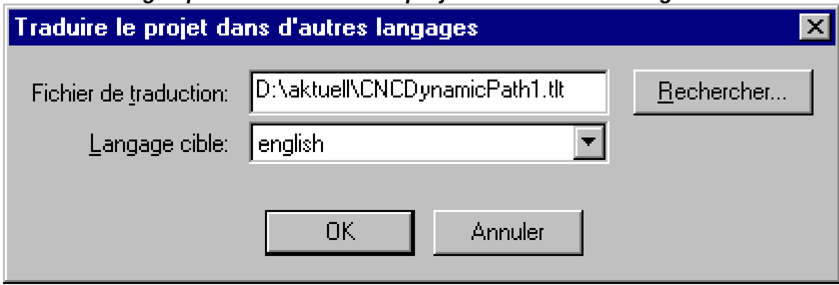
Il faut veiller à ce que les identificateurs ou noms traduits restent valables et corrects selon la norme, et que les chaînes de caractères et commentaires restent inclus entre les guillemets. Dans le cas d'un commentaire (##COMMENT_ITEM), repris dans le fichier de traduction avec "(* Commentaire 1)", le "##TODO' doit être remplacé par un "(* Commentaire 1 *)", et dans le cas d'une chaîne de caractères (##STRING_ITEM), "'chaîne de caractères1'" par "'texte1'".

Remarque : Les parties suivantes d'un fichier de traduction ne devraient pas être modifiées sans connaissances préalables approfondies : bloc linguistique, bloc de drapeaux, informations sur la position, textes originaux.

'Projet' 'Traduire projet'

Cette commande du menu 'Projet' 'Traduire dans une autre langue' active une boîte de dialogue 'Traduire le projet dans une autre langue'.

Boîte de dialogue pour la traduction du projet dans une autre langue



Le projet en cours d'utilisation peut être traduit dans une autre langue moyennant l'utilisation d'un fichier de traduction valable.

Remarque: Si vous souhaitez conserver le projet dans la version linguistique originale, faites une copie de ce projet sous un autre nom avant la traduction. **Un processus de traduction n'est pas réversible**. Veuillez noter à cet égard la possibilité d'uniquement visualiser le projet dans une autre langue, cette visualisation ne permettant pas une édition de ce projet.

Indiquez dans le champ **Fichier de traduction** le chemin d'accès du fichier de traduction à utiliser. Grâce à **Rechercher**, vous obtenez une boîte de dialogue standard Windows permettant la sélection d'un fichier.

Sous **Langue cible**, vous obtenez une liste des langues disponibles dans le fichier de traduction vous permettant de choisir la langue cible.

OK débute la traduction du projet en cours d'utilisation à l'aide du fichier de traduction indiqué dans la langue cible sélectionnée. Lors de la traduction, un dialogue d'avancement apparaît en même temps que les messages d'erreur éventuels. Après la traduction, la boîte de dialogue se referme ainsi que toutes les fenêtres d'éditeur du projet.

Annuler ferme la boîte de dialogue sans aucune modification du projet en cours d'utilisation.

Si le fichier de traduction contient des données incorrectes, un message d'erreur apparaît après avoir appuyé sur OK, reprenant le chemin d'accès du fichier et la ligne incorrecte, par exemple : "[C:\Programme\CoDeSys\projets\visu.tlt (78)]; texte de traduction attendu".

Montrer le projet traduit

Si un fichier de traduction existe pour le projet, il est possible d'afficher une des versions traduites sans pour autant écraser le projet dans sa version linguistique originale.

(Veuillez noter cette possibilité en plus de la possibilité de traduction "effective" du projet, cette dernière disposant des commandes 'Traduire le projet dans une autre langue' et 'Traduire le projet'.)

La commande 'Afficher le projet traduit' du menu 'Projet' 'Traduire dans une autre langue' donne accès au dialogue 'Montrer le projet traduit'.

Boîte de dialogue pour montrer le projet traduit

Montrer le projet traduit
Fichier de traduction: (Sans titre)
Rechercher...
Langage cible: <Langage d'origine>
<Langage d'origine>
english
german

Indiquez dans le champ **Fichier de traduction** le chemin d'accès du fichier de traduction à utiliser. Grâce à **Rechercher...**, vous obtenez une boîte de dialogue standard Windows permettant la sélection d'un fichier.

Le champ **Langue cible** vous donne une liste de sélection proposant, outre l'entrée de <Langue d'origine>, les désignations des langues contenues dans le fichier de traduction. La langue d'origine est celle qui est actuellement enregistrée avec le projet. Elle ne change que lorsque vous exécutez l'instruction 'Projet' 'Traduire'. Choisissez maintenant une des autres langues proposées puis fermez le dialogue via OK. Le projet sera alors affiché dans la langue que vous aurez sélectionnée, mais il ne sera cependant pas éditable dans cette visualisation !

Pour revenir à la langue d'origine, vous pouvez utiliser la commande 'Commuter traduction'.

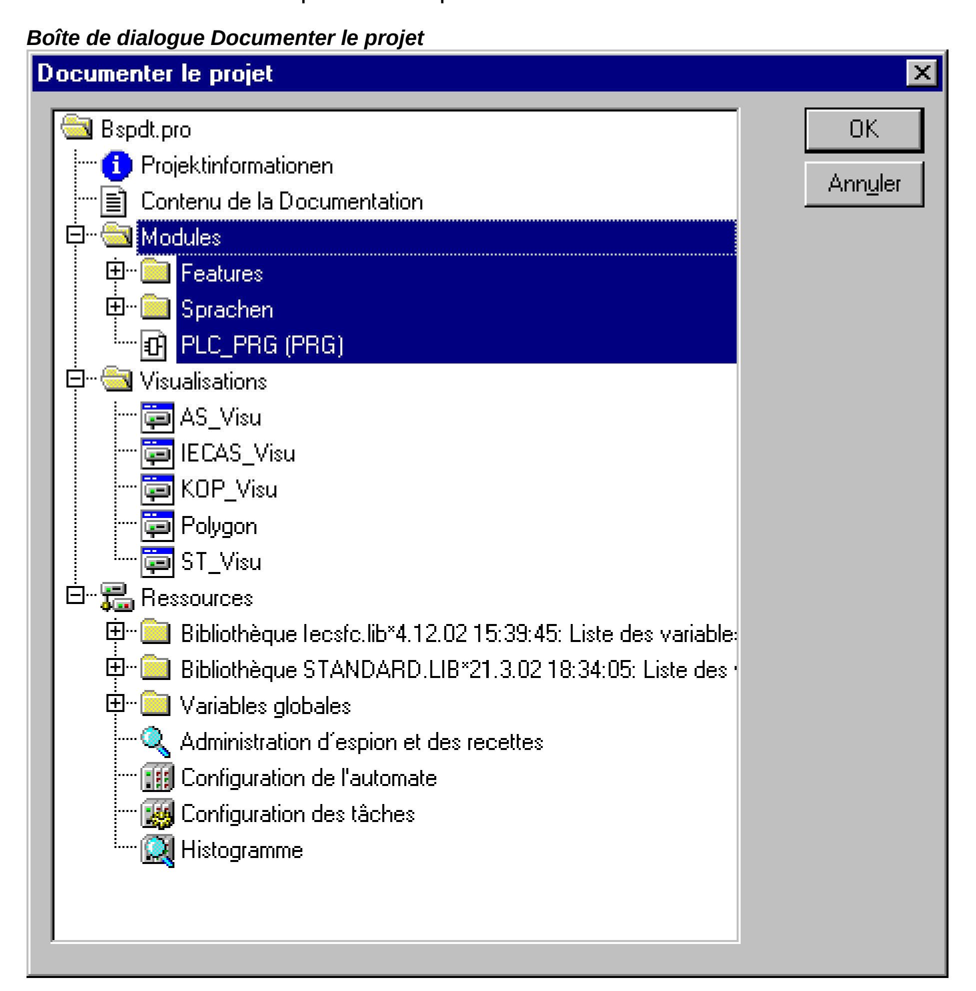
Commuter traduction

Si vous avez transformé, par le biais de la commande 'Montrer le projet traduit', l'affichage du projet (protégé en écriture) dans une autre langue disponible dans le fichier de traduction, la commande 'Commuter traduction' du menu 'Projet' 'Traduire dans une autre langue' vous permet de passer de cette version traduite à la version originale (éditable) et vice-versa.

'Projet' 'Documentation du projet'

Cette commande vous permet d'imprimer une documentation relative à l'ensemble de votre projet.

Boîte de dialogue Documenter le projet



Une documentation complète se compose des éléments suivants:

- les modules,
- le contenu de la documentation,
- les types de données,

- les visualisations,
- les ressources (variables access, variables globales, configuration des variables, histogramme, configuration de l'automate, configuration des tâches, gestionnaire d'espion et des recettes),
- les diagrammes d'appel des modules et des types de données, ainsi que
- la liste des références croisées.

Pour les deux derniers points, la compilation du projet doit avoir eu lieu sans erreur.

Les zones sélectionnées apparaissant en bleu dans la boîte de dialogue sont imprimées.

Si vous souhaitez sélectionner la totalité du projet, sélectionnez le nom du projet figurant sur la première ligne.

Par contre, si vous souhaitez sélectionner un objet unique, cliquez sur l'objet concerné ou déplacez, à l'aide des touches directionnelles, le rectangle tracé en pointillé sur l'objet souhaité. Les objets dont le symbole est précédé du caractère "+" sont des objets d'organisation contenant d'autres objets. En cliquant sur le caractère "+", l'objet d'organisation s'ouvre; en cliquant sur le caractère "–" qui apparaît alors, il se ferme. Lorsque vous sélectionnez un objet d'organisation, tous les objets qui en font partie sont sélectionnés dans le même temps. En maintenant la touche <Maj> enfoncée, vous pouvez sélectionner un bloc d'objets; la touche <Ctrl> permet de sélectionner plusieurs objets séparément.

Après avoir effectué votre sélection, cliquez sur **OK**. La boîte de dialogue Imprimer s'affiche. Vous définissez la présentation des pages à imprimer à l'aide de la commande 'Fichier' 'Configuration Documentation'.

'Projet' 'Exporter'

CoDeSys permet d'exporter ou d'importer des projets. Cette fonctionnalité vous donne la possibilité d'échanger des programmes entre différents systèmes de programmation CEI.

Jusqu'à présent, les modules en langage IL, ST et SFC ont un format d'échange standard (il s'agit du format 'Common Elements' de la norme CEI 61131-3). Pour les modules en langage LD et FBD et les autres objets, **CoDeSys** propose un format de stockage propre, car la norme CEI 61131-3 ne dispose pas de format textuel pour ces unités et objets. Les objets sélectionnés sont enregistrés dans un fichier ASCII.

Il est possible d'exporter les modules, les types de données, les visualisations et les ressources. Il est possible en outre d'exporter les entrées provenant du gestionnaire des bibliothèques, c.-à-d. les informations relatives aux liens vers les bibliothèques (pas les bibliothèques elles-mêmes !).

Attention: La réimportation d'un module FBD ou LD échoue si, dans l'éditeur graphique, un commentaire contient un guillemet simple ('), ce dernier étant interprété comme le début d'une chaîne de caractères!

Après avoir effectué votre sélection dans la boîte de dialogue, (pour la sélection, procédez comme pour 'Projet' 'Documentation du projet'), cliquez sur **OK**. La boîte de dialogue Enregistrer s'affiche. Nommez le fichier en lui attribuant l'extension ".exp".

'Projet' 'Importer'

Sélectionnez le fichier d'exportation souhaité dans la boîte de dialogue Ouvrir qui s'affiche.

Les données sont importées dans le projet actuel. S'il existe un objet du même nom dans le projet, une boîte de dialogue s'ouvre et affiche la question suivante: "Remplacer l'objet existant?": Si vous répondez **Oui**, l'objet existant est remplacé par celui du fichier d'importation; si vous répondez **Non,** le nouvel objet garde le nom de l'objet existant, lequel est complété par un caractère de soulignement et un numéro ("_0", "_1", ..). L'option **Oui, tout** ou **Non, tout** effectue la même opération pour l'ensemble des objets.

Si l'information relative à un lien vers une bibliothèque est importée, la bibliothèque en question est chargée dans le gestionnaire des bibliothèques et ajoutée à la fin de la liste. Si la bibliothèque à déjà été chargée dans le projet, elle n'est pas à nouveau chargée. Si cependant un autre moment de sauvegarde pour la bibliothèque est indiqué dans le fichier d'exportation qui est importé, le nom de la bibliothèque dans le gestionnaire des bibliothèques sera caractérisé par un "*" (p.ex. standard.lib*30.3.99 11:30:14), comme c'est le cas avec le chargement d'un projet. Si la bibliothèque

ne peut être trouvée, la boîte de dialogue ci-dessous vous informe : "Impossible de trouver la bibliothèque {<chemin d'accès>\}<nom> <date> <heure>", comme c'est le cas avec le chargement d'un projet.

La fenêtre de messages consigne l'opération.

'Projet' 'Extras' 'Importer un fichier Siemens'

Le sous-menu 'Importer un fichier Siemens' propose des commandes destinées à l'importation de blocs et de variables contenus dans les fichiers Siemens STEP5 et STEP7.

Les commandes suivantes sont proposées:

- Importer un fichier SEQ
- Importer un fichier S5

Pour plus d'informations sur l'importation de fichiers Siemens, reportez-vous à chapitre 0, 'Siemens-Import'.

'Projet' 'Comparer'

Cette commande permet de comparer deux projets, ou de comparer la version actuelle du projet ouvert avec celle qui a été enregistrée en dernier lieu.

Aperçu :

Définitions:

Projet en cours :	Projet actuellement en cours de traitement :
Projet de comparaison :	Projet appelé pour la comparaison
Mode de comparaison :	Le projet est présenté sous ce mode après sélection de la commande.
Unité :	Plus petite unité de comparaison pouvant se composer d'une ligne (Éditeur de déclaration, ST, IL), d'un réseau (FBD, LD) ou d'un élément/module (CFC,SFC).

Dans le mode de comparaison, le projet en cours et le projet de comparaison sont confrontés dans une fenêtre scindée en deux parties, et les modules étant différent sont marqués d'une couleur. Dans le cas de modules d'éditeur, les contenus sont également directement confrontés. Avant le processus de comparaison, des filtres relatifs à la prise en compte d'espaces et de commentaires peuvent être activés. Vous pouvez en outre déterminer si, en mode de comparaison, les changements intervenant à l'intérieur d'unités qui subsisteront doivent être représentés tels quels, ou si les unités différentes doivent être marquées comme 'nouveau' ou 'plus disponible'. La version du projet de comparaison peut être reprise dans le projet en cours soit par unités individuelles différentes, soit par blocs entiers identifiés comme identiques.

Veuillez noter : Tant que le mode de comparaison est activé (voir dans la barre d'état : COMPARE), le projet ne peut être édité !

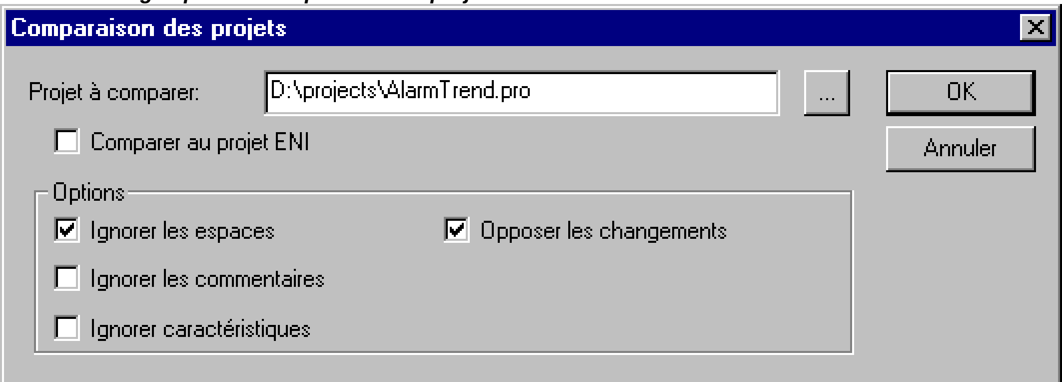
Réaliser un Comparaison de projets

Après sélection de la commande, la boîte de dialogue '**Comparaison des projets**' s'ouvre.

Introduisez le chemin d'accès du **Projet à comparer**. Le bouton ... ouvre la boîte de dialogue standard permettant l'ouverture d'un fichier que vous pouvez utiliser comme aide lors du choix du projet. Lorsque le nom du projet en cours a été saisi, la version actuelle de ce projet est comparée avec la dernière sauvegarde du même projet.

Si le projet est géré au sein d'une base de données ENI, il vous est possible de comparer la version de la base de données ouverte au niveau local avec la version actuelle de la base de données. Activez pour ce faire l'option **Comparer au projet ENI**.

Boîte de dialogue pour la comparaison de projets



Vous pouvez (dés)activer les **Options** suivantes en relation avec la comparaison :

Ignorer les espaces : Aucune différence n'est annoncée en rapport avec le nombre d'espaces.

Ignorer les commentaires :Aucune différence n'est annoncée en rapport avec les commentaires.

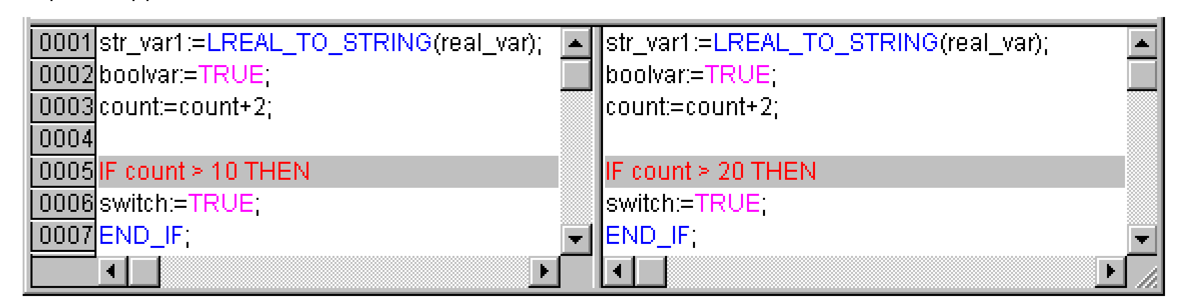
Ignorer les caractéristiques: Aucune différence n'est annoncée en rapport avec les caractéristiques des objets.

Opposer les changements :Si cette option est activée : Pour une unité au sein d'un module qui n'a pas été effacée ou ajoutée mais simplement modifiée, la version du projet de comparaison est opposée directement à la version du projet en cours grâce à la fenêtre scindée en deux du mode de comparaison (marquage en rouge, voir ci-dessous). Si cette option est désactivée : L'unité concernée est représentée comme 'plus disponible' dans le projet de comparaison et comme 'nouveau' dans le projet en cours (voir ci-dessous) et n'est donc pas directement opposée.

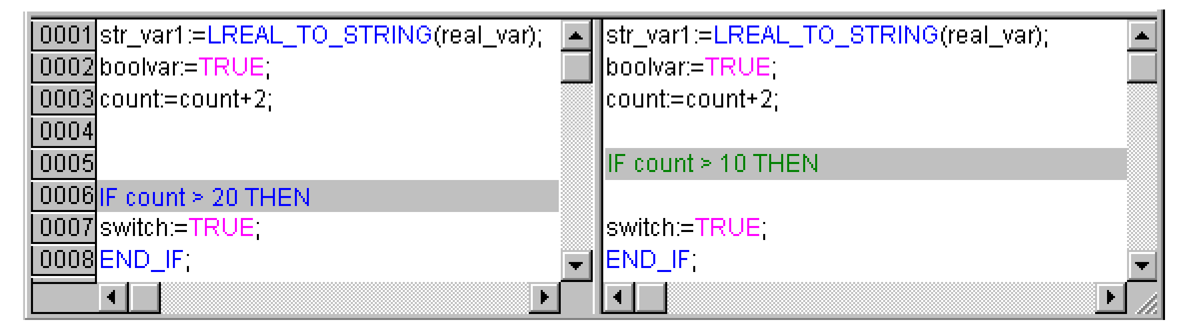
Reportez-vous à cet effet à l'exemple sur la page suivante dans lequel la ligne 0005 du projet en cours diffère (partie gauche de la fenêtre).

Exemple pour 'Opposer les changements'

Option 'Oppose differences' activated:



Option 'Oppose differences' not activated:



Lorsque vous fermez la boîte de dialogue de comparaison du projet avec OK, la comparaison s'effectue selon les réglages.

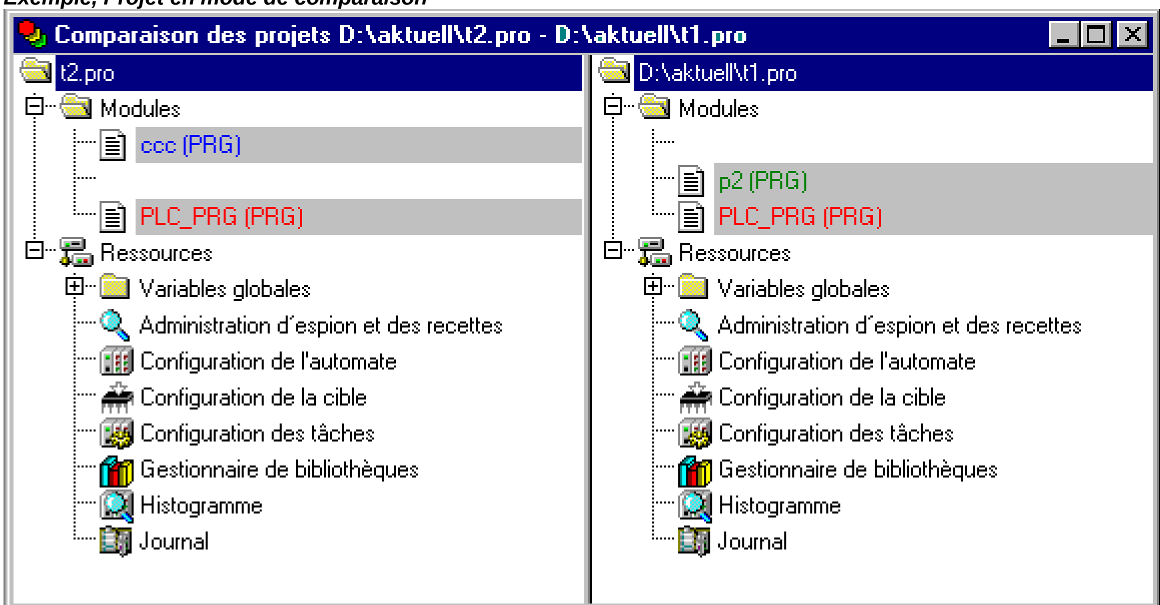
Visualisation des résultats de la comparaison

Les résultats sont tout d'abord affichés sous la forme d'une arborescence (Aperçu du projet) à partir de laquelle des modules individuels peuvent être ouverts de façon à visualiser les différences au niveau du contenu.

1. Aperçu du projet en mode de comparaison :

Lorsque la procédure de comparaison des projets est terminée, une fenêtre scindée en deux parties s'ouvre, représentant l'arborescence du projet en **mode de comparaison**. Vous pouvez lire dans la barre de titre : "Comparaison de projets <Chemin d'accès du projet en cours> - <Chemin d'accès du projet de comparaison>". La partie gauche de la fenêtre affiche le projet en cours, la partie droite le projet de comparaison. L'aperçu du projet indique en position supérieure le nom du projet et correspond par ailleurs à la structure de l'Organisateur d'objets :

Exemple, Projet en mode de comparaison



Les modules qui présentent des différences sont affichés ombrés et sont caractérisés par la couleur du texte ou par l'ajout d'un texte :

Rouge : une unité qui a été modifiée est représentée en rouge dans les deux parties de la fenêtre.

Bleu : unité disponible uniquement dans le projet en cours; un vide est introduit à ce même emplacement dans l'arborescence du projet de comparaison.

Vert : unité disponible uniquement dans le projet de comparaison ; un vide est introduit à ce même emplacement dans l'arborescence du projet en cours.

Noir : unité pour laquelle aucune différence n'a été constatée.

"**(Propriétés changées)**" : Ce texte apparaît derrière le nom d'un module dans l'arborescence du projet en cours lorsque des différences ont été constatées dans les propriétés du module.

"**(Droits d'accès modifiés)**" : Ce texte apparaît derrière le nom d'un module dans l'arborescence du projet en cours lorsque des différences ont été constatées dans les droits d'accès.

2. Contenu des modules en mode de comparaison :

Vous **ouvrez le module** concerné à l'aide d'un double-clic sur la ligne dans l'aperçu du projet.

S'il s'agit d'un module texte ou graphique modifié (rouge), il sera ouvert dans une fenêtre scindée en deux parties. Le contenu du module dans le projet de comparaison (à droite) sera opposé à celui du projet en cours, comme c'était le cas dans l'aperçu du projet. Les différences constatées seront représentées avec les mêmes couleurs que celles utilisées ci avant.

Exemple de module en mode de comparaison

PLC_PRG (PRG-ST)

```
PROGRAM PLC_PRG
VAR
    count: INT;
    switch: BOOL;
    boolvar: BOOL;
    str_var1: STRING(10);
    str_var2: STRING(18);
    real_var: LREAL;
    count1: INT;
END_VAR
```

```
PROGRAM PLC_PRG
VAR
    count: INT;
    switch: BOOL;
    boolvar: BOOL;
    str_var1: STRING(10);
    str_var2: STRING(18);
    real_var: LREAL;

END_VAR
```

```
str_var1:=LREAL_TO_STRING(real_var);
str_var2:=TIME_TO_STRING(T#12ms);
boolvar:=FALSE;
count:=count+2;

IF count > 20 THEN
switch:=TRUE;
END_IF;






count1:=count1+1;
```

```
str_var1:=LREAL_TO_STRING(real_var);
str_var2:=TIME_TO_STRING(T#12ms);
boolvar:=TRUE;
count:=count+2;

IF count > 20 THEN
switch:=TRUE;
END_IF;
IF count > 40 THEN
switch:=FALSE;
END_IF;
IF count = 50 THEN
count:=0;
END_IF;
```

cette ligne a été modifiée (rouge)

ces lignes ont été effacées (bleu)

cette ligne vient d'être insérée (vert)

S'il ne s'agit pas d'un module éditeur, mais bien par exemple d'une configuration de tâche, de configurations du système cible, etc., le module du projet en cours ou le module du projet de comparaison sera ouvert dans une propre fenêtre selon que vous cliquiez sur la partie gauche ou droite de l'aperçu du projet. Pour ces modules de projet, il n'y a pas d'autre différenciation au niveau du contenu.

Travailler en mode de comparaison

Si, dans la fenêtre scindée en deux parties, le curseur se trouve sur une ligne indiquant une différence, le menu **'Extras'** ou le **menu contextuel** (bouton droit de la souris) offre une sélection des commandes suivantes, selon que l'on se trouve au niveau de l'aperçu du projet ou à l'intérieur d'un module :

- `Prochaine différence'
- `Différence précédente'
- `Accepter les parties changées'
- `Accepter élément changé'
- `Accepter les caractéristiques'
- 'Accepter les droits d'accès'

'Extras' 'Prochaine différence'

Icône : <F7>

Le curseur se déplace jusqu'à l'emplacement suivant (ligne dans l'aperçu du projet/ligne ou réseau dans le module) indiquant une différence.

'Extras' 'Différence précédente'

Icône : <Maj><F7>

Le curseur se déplace vers l'emplacement précédent (ligne dans l'aperçu du projet/ligne ou réseau dans le module) indiquant une différence.

'Extras' 'Accepter les parties changées'

Icône : <Ctrl> <Espace> : Accepter les parties changées ou **Accepter changement**

Pour toutes les unités allant ensemble (p.ex. des lignes consécutives) et étant caractérisées par la même modification, la version du projet de comparaison est reprise dans le projet en cours. Les unités concernées apparaissent alors dans la partie gauche de la fenêtre dans la couleur adéquate. S'il s'agit d'une unité qui était marquée en rouge (modification à l'intérieur), la reprise dans le projet en cours est indiquée par la couleur jaune.

'Extras' 'Accepter élément changé'

Icône : <Ctrl> <Espace> : (uniquement dans l'aperçu du projet)

La version du projet de comparaison n'est reprise dans le projet en cours que pour l'unité de comparaison sur laquelle le curseur se trouve actuellement (p.ex. ligne dans l'aperçu du projet, ou ligne/réseau dans les modules). L'unité concernée apparaît alors dans la partie gauche de la fenêtre dans la couleur adéquate. S'il s'agit d'une unité qui était marquée en rouge (modification à l'intérieur), la reprise dans le projet en cours est indiquée par la couleur jaune.

'Extras' 'Accepter les caractéristiques'

(uniquement dans l'aperçu du projet)

Pour le module sur lequel le curseur se trouve actuellement, les caractéristiques du module provenant du projet de comparaison seront reprises dans le projet en cours.

'Extras' 'Accepter les droits d'accès '

(uniquement dans l'aperçu du projet)

Pour le module sur lequel le curseur se trouve actuellement, les droits d'accès du module provenant du projet de comparaison seront reprises dans le projet en cours.

'Projet' 'Copier'

Cette commande permet de copier dans votre projet des objets (modules, types de données, visualisations et ressources) ainsi que des liens vers des bibliothèques contenus dans d'autres projets.

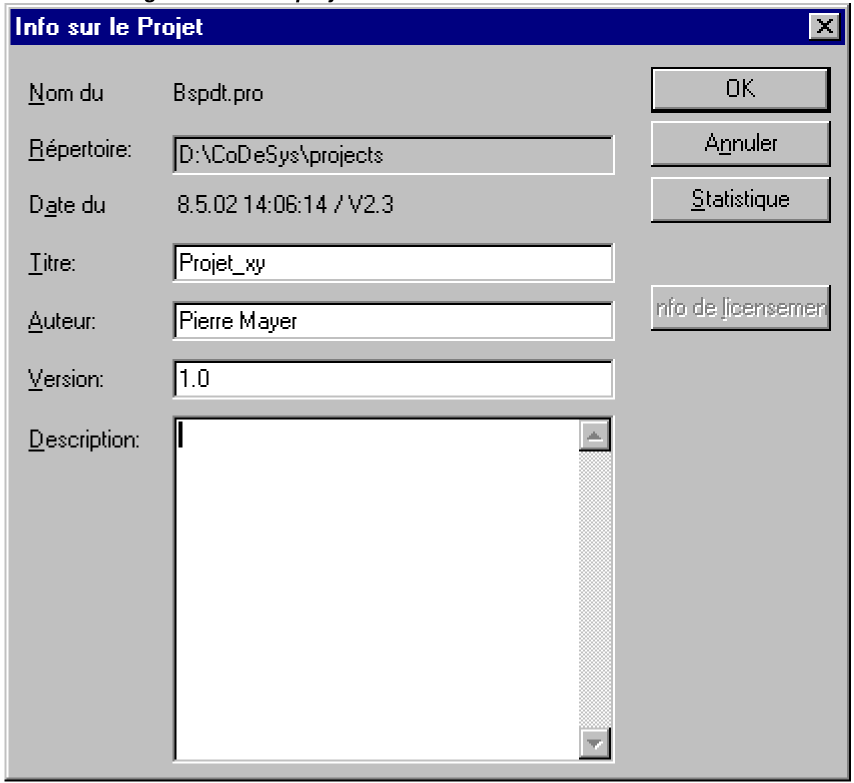
Lorsque vous sélectionnez cette commande, la boîte de dialogue standard Ouvrir s'affiche, si vous avez sélectionné un fichier, puis une autre boîte de dialogue s'ouvre, dans laquelle vous pouvez sélectionner les objets souhaités. La sélection est effectuée de la même façon que pour 'Projet' - '**Documentation du projet**'.

S'il existe déjà un objet portant le même nom dans le projet, le nouvel objet garde le nom de l'objet existant, lequel est complété par un caractère de soulignement et un numéro ("_1", "_2" ...).

'Projet' 'Informations sur le projet'

Grâce à cette option, vous pouvez mémoriser des informations relatives à votre projet. Cette commande donne accès à la boîte de dialogue suivante:

Boîte de dialogue Info sur le projet



Les données suivantes sont les informations relatives au projet:

- Nom de fichier
- Répertoire (avec chemin)
- Date de la dernière modification (Date du dernier changement)
- Ces données ne peuvent pas être modifiées.
- En outre, vous pouvez entrer diverses informations:
- un **Titre** pour le projet
  Veuillez noter : Pour autant que cela soit supporté par le système cible, la désignation saisie ici est automatiquement proposée comme nom de fichier dès que le projet est à nouveau chargé dans CoDeSys via la fonction 'Fichier' 'Ouvrir' 'Ouvrir le projet de l'automate programmable' (provoquant dans ce cas l'ouverture du dialogue d'enregistrement de fichier).
- le nom de l'**Auteur**,
- le numéro de la **Version**, et
- une **Description** du projet.

Ces données sont optionnelles. Lorsque vous cliquez sur le bouton **Statistique**, vous obtenez une description statistique du projet.

Cette description contient les données reprises dans la boîte de dialogue **Info sur le projet**, ainsi que le nombre de **Modules**, de **Types de données**, de **Variables locales** et de **Variables globales**, tels qu'ils ont été enregistrés lors de la dernière compilation.

Exemple de statistique de projet

Statistique du Projet

Nom du ampel.pro
Répertoire: C:\CoDeSys\Projects
Date du 22.12.98 19:13:35
Titre:
Auteur:
Version:
Statistique
Unités de 5
Types de 0
Variables Globales: 5
Variable Locales: 43
L'état de la dernière compilation est affiché.
Fermer

Vous pouvez appuyer sur le bouton **Infos relatives à la licence ...** s'il s'agit d'un projet CoDeSys qui a déjà été enregistré comme module soumis à licence par le biais de la commande 'Fichier' 'Enregistrer sous...'. Vous obtenez dans ce cas la boîte de dialogue de 'Editer informations relatives à l'attribution d'une licence', dans laquelle vous pouvez modifier ou effacer les informations relatives à cette licence. Voyez à cet effet : 'Gestion des licences dans CoDeSys'.

Si vous sélectionnez l'option **Demander de l'information sur le projet** dans la catégorie Ouvrir & Fermer de la boîte de dialogue Options, les informations sur le projet sont automatiquement appelées lors de l'enregistrement d'un nouveau projet ou de l'enregistrement d'un projet sous un nouveau nom.

'Projet' 'Recherche globale'

Cette commande vous permet de rechercher un texte dans les modules, les types de données ou les objets des variables globales.

Lorsque la commande a été appelée, une boîte de dialogue s'affiche, dans laquelle vous pouvez sélectionner les objets souhaités. La sélection est effectuée de la même façon que pour 'Projet' - 'Documentation du projet'.

Après avoir cliqué sur **OK** pour confirmer votre choix, la boîte de dialogue Rechercher s'affiche. Si un texte est trouvé dans un objet, ce dernier est chargé dans l'éditeur auquel il est lié et le lieu où il a été découvert est affiché. Pour ce qui est de l'affichage du texte trouvé ainsi que de la recherche et de la poursuite de la recherche, ils sont analogues à la commande 'Rechercher' du menu 'Editer'.

Après avoir cliqué sur **OK** pour confirmer votre choix, la boîte de dialogue standard permettant la recherche s'affiche. Celle-ci s'ouvre directement si la commande 'Recherche globale' a été appelée via le symbole dans la barre des menus, et la recherche se rapporte automatiquement à toutes les parties du projet pouvant faire l'objet d'une recherche. Les dernières recherches introduites peuvent être sélectionnées via la zone de liste modifiable du champ **Rechercher**. Si un texte est trouvé dans un objet, ce dernier est chargé dans l'éditeur ou le gestionnaire de bibliothèques auquel il est lié et le lieu où il a été découvert est affiché. Pour ce qui est de l'affichage du texte trouvé ainsi que de la recherche et de la poursuite de la recherche, ils sont analogues à la commande "Rechercher" du menu "Editer".

Si vous sélectionnez le bouton **Fenêtre de messages**, tous les endroits où la suite de caractères recherchée est utilisée dans les objets sélectionnés sont énumérés ligne par ligne sous forme de tableau dans une fenêtre de messages. Le nombre d'occurrences trouvées est donné à la fin.

Si la fenêtre de messages n'était pas ouverte, elle est affichée. Par emplacement trouvé, les données suivantes sont affichées :

- Nom d’objet
- Emplacement dans la partie de déclaration (Décl) ou dans la partie d’implémentation (Impl) d’un module
- Numéro de ligne ou de réseau
- Ligne complète dans le cas des éditeurs littéraux
- Unité de texte complète dans le cas des éditeurs graphiques

Fenêtre des messages avec affichage des résultats de recherche

```
Recherche globale: '250'
ST_EXAMPLE (PRG-ST)(Déclaration) #6 yVal: DINT := -250;
ST_EXAMPLE (PRG-ST)(Déclaration) #7 bottom: DINT := -250;
ST_EXAMPLE (PRG-ST)(Implémentation) #20 IF (bottom > -250) THEN
ST_EXAMPLE (PRG-ST)(Implémentation) #42        yVal := -250;
Nombre de lignes trouvées: 4
```

Si, dans la fenêtre de messages, vous exécutez un double-clic sur une ligne avec la souris ou si vous enfoncez la touche <Entrée>, alors l’éditeur de l’objet s’ouvre. La ligne concernée de l’objet est marquée. Vous pouvez parcourir rapidement les lignes d’affichage au moyen de la touche de fonction <F4> ou <Maj> + <F4>.

'Projet' 'Remplacer globalement'

Cette commande vous permet de rechercher un texte dans les modules, les types de données ou les objets des variables globales et de le remplacer par un autre. Pour ce qui est de la manipulation et de la procédure, elles sont identiques à celles décrites dans la partie 'Projet' - 'Recherche globale' ou 'Editer' - 'Remplacer'. Cependant, une sélection des bibliothèques n’est pas proposée et il n’y a pas d’affichage possible dans la fenêtre de messages.

'Projet' 'Vérifier tout'

Grâce à cette commande, vous pouvez ouvrir un sous-menu comprenant les commandes suivantes permettant de contrôler l'exactitude sémantique du projet :

- Variables inutilisées
- Zones de mémoire superposées
- Accès concurrent
- Ecriture multiple sur la sortie

Les résultats seront affichés dans une fenêtre de messages.

Chacune de ces fonctions contrôle l'état de la dernière compilation. Le projet doit donc avoir été compilé au moins une fois sans erreurs avant d'effectuer un contrôle, sans quoi les options de menu s'en trouvent « inhibées ».

Remarque: Ces contrôles peuvent également être configurés dans les options de projet, catégorie Options de compilation, de telle sorte qu'ils soient automatiquement effectués à chaque compilation.

Variables inutilisées

Cette fonction recherche des variables qui sont déclarées mais non utilisées dans le programme. Elles sont affichées avec le nom du module et la ligne, par exemple : PLC_PRG (4) – var1. Les variables contenues dans des bibliothèques ne sont pas prises en compte.

Zones de mémoire imbriquées

Cette fonction vérifie s’il n’y a pas de recoupements dans certaines zones de mémoire lors de l’affectation de variables via une déclaration AT. À titre d’exemple, il y a un recoupement de par

l'affectation des variables "var1 AT %QB21: INT" et "var2 AT %QD5: DWORD", puisqu'elles ont l'octet 21 de commun. L'affichage ressemble alors à ceci :

%QB21 est référencé par les variables suivantes:
PLC_PRG (3): var1 AT %QB21
PLC_PRG (7): var2 AT %QD5

Accès concurrent

Cette fonction recherche des zones de mémoire d'adresses CEI qui sont référencées dans plus d'une tâche. Il n'est pas fait de différence entre l'accès en lecture ou l'accès en écriture. L'affichage ressemble à ceci :

%MB28 est référencé dans les tâches suivantes :
Task1 - PLC_PRG (6): %MB28 [accès lecture seule]
Task2 - POU1.ACTION (1) %MB28 [accès en écriture]

Ecriture multiple sur la sortie

Cette fonction recherche des zones de mémoire auxquelles on accède en écriture à partir de plusieurs endroits au sein d'un projet. À titre d'exemple, l'affichage ressemble à ceci :

%QB24 est accédé en écriture aux emplacements suivants :
PLC_PRG (3): %QB24
PLC_PRG.POU1 (8): %QB24

Niveaux d'accès

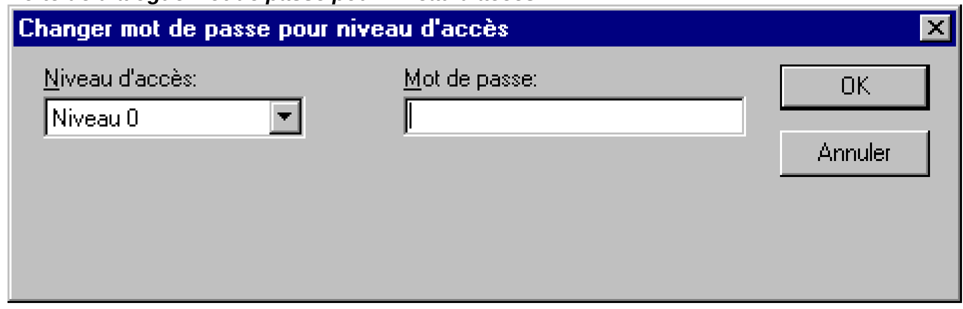
Le système **CoDeSys** permet de configurer jusqu'à huit niveaux d'accès pour les modules, types de données, visualisations et ressources. Les droits d'accès peuvent être accordés pour la totalité des objets ou pour certains d'entre eux. A chaque fois qu'un projet est ouvert, cette ouverture se fait avec un niveau d'accès déterminé. Ce niveau d'accès est protégé par un mot de passe.

Les niveaux d'accès sont numérotés de 0 à 7, le niveau 0 détenant les droits de l'administrateur, c.-à-d. ceux de définir les mots de passe et les droits d'accès pour tous les niveaux ou objets.

Lors de la création d'un nouveau projet, aucun mot de passe n'est défini. Tant qu'aucun mot de passe n'a été défini pour le niveau 0, l'utilisateur entrant dans le projet est automatiquement considéré comme utilisateur appartenant au niveau 0.

Lorsqu'un mot de passe est défini pour le niveau 0 au moment du chargement du projet, *tous les* niveaux doivent introduire un mot de passe pour ouvrir le projet concerné. Pour cela, la boîte de dialogue suivante s'affiche:

Boîte de dialogue Mot de passe pour niveau d'accès



Indiquez le niveau auquel vous appartenez dans la zone de liste modifiable **Niveau d'accès** située dans la partie gauche de la boîte de dialogue, et tapez le Mot de passe correspondant dans la partie droite de la fenêtre. Cliquez sur **OK**. Lorsque le mot de passe ne concorde pas avec celui mémorisé, le message suivant s'affiche:

`„Le mot de passe est incorrect."`

Le projet ne s'ouvre qu'avec le mot de passe correct.

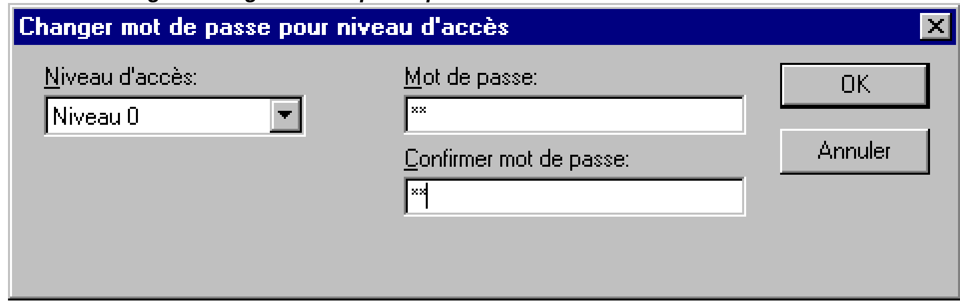
Attention : Si des mots de passe ne sont pas attribués à tous les groupes de travail, un projet peut être ouvert par un groupe de travail auquel aucun mot de passe n'a été attribué.

La commande 'Mots de passe pour niveau d'accès' vous permet d'attribuer des mots de passe et la commande 'Objet' 'Droits d'accès' attribue les droits pour les objets individuels ou pour la totalité d'entre eux.

'Projet' 'Mot de passe pour niveau d'accès'

Cette commande permet d'ouvrir la boîte de dialogue Changer mot de passe pour niveau d'accès. Il n'est accessible qu'aux membres du niveau 0. Cette commande donne accès à la boîte de dialogue suivante:

Boîte de dialogue Changer mot de passe pour niveau d'accès



Vous pouvez sélectionner le niveau dans la zone de liste modifiable Niveau d'accès située dans la partie gauche de la fenêtre. Tapez le mot de passe correspondant au niveau choisi dans la zone Mot de passe. Une étoile (*) apparaît à chaque fois que vous saisissez une lettre. Confirmez-le en le tapant à nouveau dans la zone **Confirmer mot de passe**. Fermez ensuite la boîte de dialogue en cliquant sur **OK**. Si le message

„Le mot de passe et la confirmation ne concordent pas."

apparaît, vous avez fait une faute de frappe au cours d'une des entrées. C'est pourquoi il est préférable de répéter les deux saisies jusqu'à ce que la boîte de dialogue se ferme sans qu'un message n'apparaisse.

Vous pouvez ensuite éventuellement saisir un mot de passe pour le niveau inférieur en appelant à nouveau la commande.

Il est possible d'attribuer des droits pour les objets individuels ou tous les objets à l'aide de 'Objet' 'Droits d'accès'

Attention : Si des mots de passe ne sont pas attribués à tous les groupes de travail, un projet peut être ouvert par un groupe de travail auquel aucun mot de passe n'a été attribué.

La commande 'Mots de passe pour niveau d'accès' vous permet d'attribuer des mots de passe et la commande 'Objet' 'Droits d'accès' attribue les droits pour les objets individuels ou pour la totalité d'entre eux.

'Projet' 'Liaison à la base de données'

Cet élément de menu est disponible lorsque l'option 'Utiliser la base de données du projet (ENI)' des options du projet, catégorie Base de données du projet est activée. Elle mène à un sous-menu avec les commandes pour la gestion de l'objet ou du projet dans la base de données en cours d'utilisation reliée via l'interface ENI.

- Login (ouverture de session de l'utilisateur auprès du serveur ENI)

Lorsqu'un objet est marqué dans l'Organisateur d'objets et que la commande 'Liaison à la base de données' est sélectionnée du **menu contextuel** (bouton droit de la souris), les fonctions correspondantes de la base de données peuvent être appelées pour cet objet par le biais des commandes présentées. Au cas où aucune ouverture de session n'a été faite pour l'utilisateur auprès de ENI via la boîte de dialogue **Ouverture de session-Base de données**, cette dernière s'ouvre tout d'abord et la commande n'est exécutée qu'après authentification de l'utilisateur.

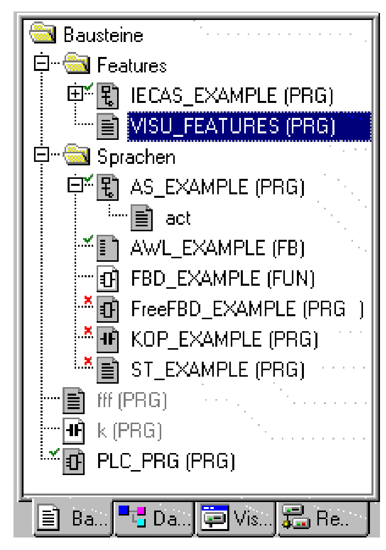
- Déinir
- Dernière version

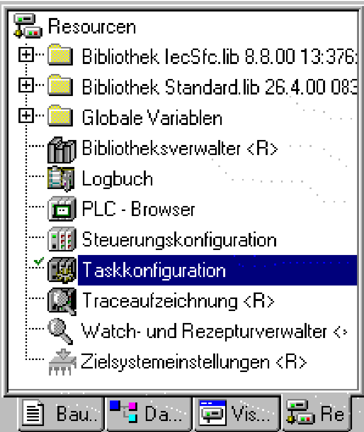
- Check out
- Check in
- Annuler check out
- Afficher les différences
- Afficher l'histoire de la version

Si la commande 'Liaison à la base de données' a été sélectionnée dans le **Menu 'Projet'**, d’autres options de menu s’affichent, relatives à tous les objets du projet :

- Définition multiple
- Dernières versions
- Check out multiple
- Check in multiple
- Annuler le check-out multiple
- Historique de la version du projet
- Attribuer un nom à la version
- Insérer les objets commun
- Actualiser le statut

Présentation des états des objets relatifs à la gestion dans la base de données du projet dans l’Organisateur d’objets :





Icône ombré en gris :

l’objet est géré dans la base de données.

Crochet vert devant le nom de l’objet :

check-out de l’objet hors du projet CoDeSys ouvert actuellement.

Croix rouge devant le nom de l’objet :

check-out actuel de l’objet par un autre utilisateur.

<R> après le nom de l’objet :

vous ne pouvez accéder à l’objet qu’en lecture seule.

Veuillez noter : Certains objets (configuration des tâches, configuration de l’histogramme, configuration de l’automate, configuration du système cible, gestionnaire d’espion et de recettes) sont en principe munis d’un <R> tant qu’ils n’ont pas encore subi de check-out. Ceci signifie qu’il n’y aura pas la question automatique "Check-out de l’objet ?" quand on débute l’édition de l’objet; cela ne veut pas nécessairement dire qu’un accès en écriture est interdit. Le fait que cet accès ne soit pas possible est reconnaissable au fait que le commande de check-out n’est pas sélectionnable.

Définir

Commande : 'Projet' 'Liaison avec la base de données' 'Définir'

Nous allons déterminer si l'objet marqué dans l'Organisateur d'objets doit être géré dans la base de données ou uniquement localement (au sein du projet). À cet effet, une boîte de dialogue s'ouvre dans laquelle vous pouvez choisir entre les deux catégories de base de données 'Projet' ou 'Objets communs', ou la catégorie 'Local'. Voir également à cet effet Catégories au sein de la base de données du projet.

Les icônes de tous les objets qui sont gérés dans la base de données apparaissent ombrés en gris dans l'Organisateur d'objets.

Dernière version

Commande : 'Projet' 'Liaison avec la base de données' 'Dernière version'

La version actuelle de l'objet marqué dans l'Organisateur d'objets est extraite de la base de données et remplace la version locale. Contrairement au check-out - voir ci-dessous -, l'objet dans la base de données n'est pas bloqué au traitement par d'autres utilisateurs.

Check out

Commande : 'Projet' 'Liaison avec la base de données' 'Check out'

L'objet marqué dans l'Organisateur d'objets subit un check-out hors de la base de données et est alors bloqué au traitement par d'autres utilisateurs.

A l'appel de cette commande, la boîte de dialogue 'Extraire objet' s'ouvre. Vous pouvez introduire un commentaire qui sera enregistré dans l'historique de l'objet dans la base de données en même temps que la procédure de check-out.

Après avoir confirmé la boîte de dialogue avec OK, l'objet ayant subi un check-out dans l'Organisateur d'objets est caractérisé par la présence d'un crochet devant le nom du module, il apparaît pour les autres utilisateurs marqué d'une croix rouge et ne peut donc pas être traité par ces autres utilisateurs.

Check in

Commande : 'Projet' 'Liaison avec la base de données' 'Check in'

L'objet marqué dans l'Organisateur d'objets subit un check-in dans la base de données. Une nouvelle version de l'objet est ainsi créée dans la base de données. Les anciennes versions sont conservées.

A l'appel de cette commande, la boîte de dialogue 'Archiver l'objet' s'ouvre. Vous pouvez introduire un commentaire qui sera enregistré dans l'historique de l'objet dans la base de données en même temps que la procédure de check-in.

Après la confirmation de la boîte de dialogue avec OK, le crochet vert devant le nom du module disparaît de l'Organisateur d'objets.

Annuler check out

Commande : 'Projet' 'Liaison avec la base de données' 'Annuler check out'

Le check-out de l'objet marqué dans l'Organisateur d'objets et les changements effectués localement au sein de cet objet sont annulés. Aucune boîte de dialogue n'apparaît. L'objet demeure inchangé et est à nouveau libéré pour les autres utilisateurs dans la base de données. La croix rouge devant le nom du module dans l'Organisateur d'objets disparaît.

Afficher les différences

Commande : 'Projet' 'Liaison avec la base de données' 'Afficher les différences'

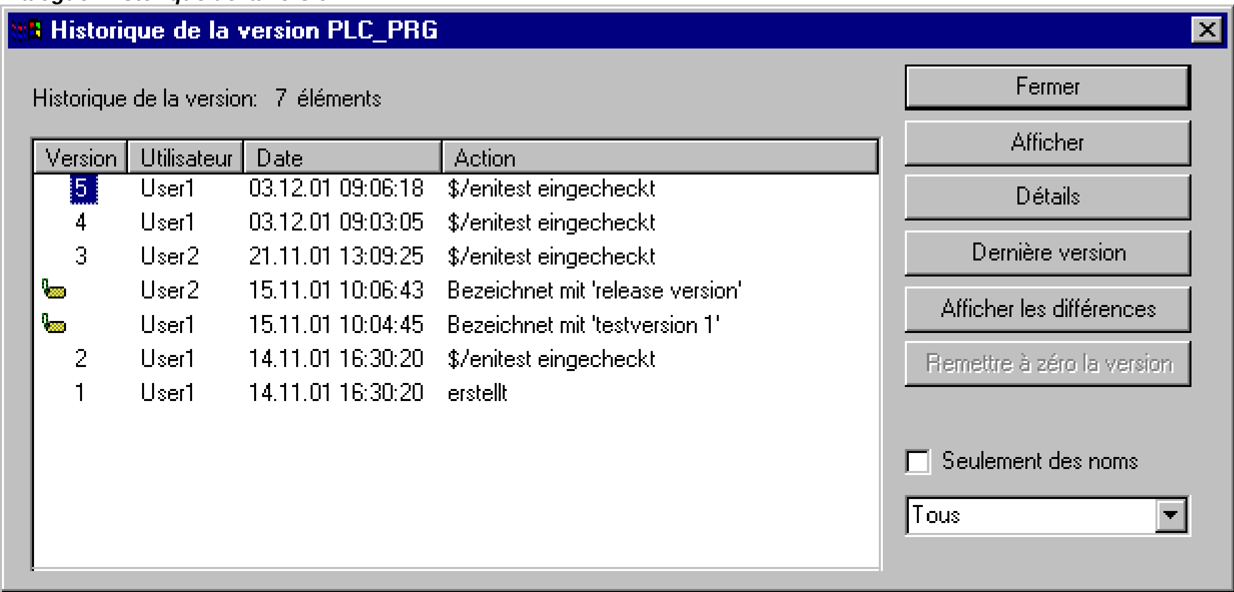
Le module en cours de traitement dans CoDeSys est présenté en une fenêtre scindée en deux parties opposant la version locale en cours de traitement à la dernière version actuelle de la base de données. Les différences entre les deux versions sont représentées visuellement comme lors de la comparaison de projets (voir 'Projet' 'Comparer').

Afficher l'histoire de la version

Commande : 'Projet' 'Liaison avec la base de données' 'Afficher l'histoire de la version'

Une boîte de dialogue 'Historique de la version <Nom d'objet>' s'ouvre, énumérant toutes les versions de l'objet en cours de traitement et qui ont fait l'objet d'un check-in ou ont été étiquetées. Cette boîte de dialogue comprend :

Dialogue 'Historique de la version



Version : Numérotation des versions de l'objet ayant subi un check-in, par ordre chronologique et en fonction de la base de données. Les versions étiquetées ne reçoivent pas de numéro de version, mais sont caractérisées par une icône d'étiquette.

Utilisateur : nom de l'utilisateur qui a effectué l'action sur l'objet

Date : date et heure de l'action

Action : type d'action qui a été effectuée sur l'objet. Dépendant de la base de données, par exemple 'créé' (l'objet a subi un premier check-in dans la base de données), 'check-in' ou ‚étiqueté avec <label>' (cette version de l'objet a été pourvue d'un identificateur)

Les boutons :

- **Fermer** : la boîte de dialogue se referme
- **Afficher** : la version marquée dans le tableau est ouverte dans une fenêtre dans CoDeSys. Vous pouvez lire dans la barre de titre : ENI: <Nom du projet dans la base de données>/<Nom d'objet>
- **Détails** : La boîte de dialogue 'Détails de l'historique de la version' s'ouvre :
- **Fichier, Nom** (répertoire du projet et du nom de l'objet dans la base de données), **Version** (voir ci-dessus), **Date** (voir ci-dessus), **Utilisateur** (voir ci-dessus), **Commentaire** (commentaire introduit lors du check-in ou de l'étiquetage). Les boutons **Prochain** ou **Précédent** permettent de sauter aux détails de l'entrée précédente ou suivante dans le dialogue 'Historique de la version...'.
- **Dernière version** : La version marquée dans le tableau est chargée de la base de données de CoDeSys et remplace la version locale.
- **Afficher les différences :** Si une seule version de l'objet est marquée dans le tableau, la commande a pour effet de comparer cette version avec la version actuelle de la base de données. Si deux versions sont marquées, celles-ci sont comparées. Les différences sont représentées dans une fenêtre scindée en deux parties comme avec la comparaison de projets.
- **Remettre à zéro la version** : la version marquée dans le tableau est établie comme version actuelle de la base de données. Les versions introduites ultérieurement sont effacées ! Ceci peut être utilisé pour rétablir une version antécédente et l'établir comme version actuelle.

- **Seulement des noms** : Si cette option est activée, seules les versions pourvues d'une étiquette apparaissent à la sélection dans le tableau.
- **Boîte de sélection** sous l'option 'Seulement des noms' : Les noms de tous les utilisateurs ayant déjà effectué des actions de base de données sur les objets du projet sont repris ici. Sélectionnez 'Tous' ou un seul des noms afin de visualiser l'historique de version des objets traités soit par tous les utilisateurs soit par un utilisateur particulier.

Définition multiple

Commande : 'Projet' 'Liaison avec la base de données' 'Définition multiple'

À l'aide de cette commande, vous pouvez déterminer, pour plusieurs objets du projet en cours d'utilisation en même temps, dans quelle catégorie de base de données ils seront gérés. La même fenêtre de dialogue '**Propriétés de l'objet**' que pour la commande 'Définir' apparaît tout d'abord. Choisissez-y la catégorie souhaitée et fermez cette boîte de dialogue avec OK. Suite à quoi la boîte de dialogue **'Sélection ENI'** s'ouvre, énumérant les modules du projet pertinents pour la catégorie concernée (par exemple, si vous avez choisi la catégorie 'Ressources', les modules ressources du projet seront proposés à la sélection). La représentation correspond à l'arborescence utilisée pour l'Organisateur d'objets. Marquez les modules souhaités et confirmez avec **OK**.

Dernières versions

Commande : 'Projet' 'Liaison avec la base de données' 'Dernières versions'

Pour le projet ouvert, la version actuelle de tous les objets de la catégorie projet sont prises de la base de données. Si des objets ont été ajoutés dans la base de données, ceux-ci seront également introduits localement, et si des objets ont été effacés de la base de données, ceux-ci ne seront pas effacés localement, mais bien classés automatiquement dans la catégorie 'Local'. Pour les objets provenant de la catégorie Ressources, ne sont prises (appelées) de la base de données que les objets qui sont déjà créés dans le projet local. Pour la signification de l'appel, reportez-vous à la commande 'Dernière version'

Check out multiple

Commande : 'Projet' 'Liaison avec la base de données' 'Check out multiple'

Plusieurs objets peuvent en même temps faire l'objet d'un check-out. À cet effet, la boîte de dialogue **'Sélection ENI'** s'ouvre, énumérant les modules du projet comme dans l'arborescence de l'Organisateur d'objets. Marquez les modules qui doivent faire l'objet d'un check-out, et confirmez avec OK. Pour la signification du check-out, reportez-vous à la commande 'Check out'.

Check in multiple

Commande : 'Projet' 'Liaison avec la base de données' 'Check in multiple'

Plusieurs objets peuvent en même temps faire l'objet d'un check-in. La procédure est la même que pour le check-out multiple. Pour la signification du check-in, reportez-vous à la commande 'Check in'.

Annuler le check-out multiple

Commande : 'Projet' 'Liaison avec la base de données' 'Annuler le check-out multiple'

Vous pouvez annuler le Check-out pour plusieurs objets en même temps. La sélection s'effectue comme pour le 'Check-out multiple' ou le 'Check in multiple'.

Historique de la version du projet

Commande 'Projet' 'Liaison avec la base de données' 'Historique de la version du projet'

Choisissez cette commande pour visionner l'historique de la version du projet en cours.

Vous obtenez alors la boîte de dialogue 'Historique de la version du <Nom du projet dans la base de données>', reprenant une liste chronologique de toutes les actions (création, check-in, étiquetage) pour tous les objets appartenant au projet. Le nombre de ces objets est indiqué derrière l'**Historique de la version du projet**. Pour l'utilisation du dialogue, voir ci-dessous sous le point 'Afficher l'histoire de la version', pour les objets individuels notez cependant ce qui suit :

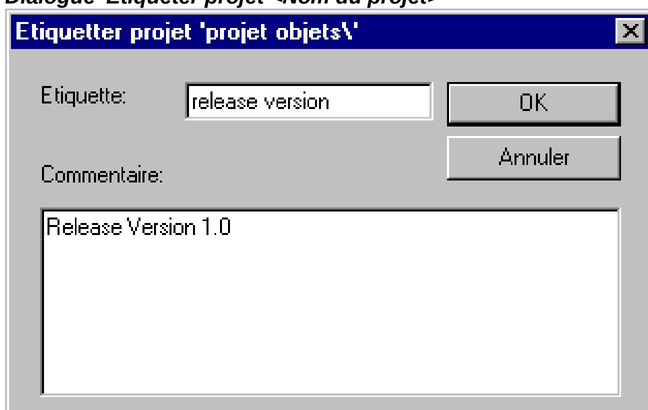
- La commande 'Remettre à zéro la version' ne s'applique qu'à des objets individuels
- La commande 'Dernière version' signifie que tous les objets provenant de la version du projet marquée au tableau sont reprises dans le projet local. Ceci a pour effet que les objets locaux sont écrasés avec des versions précédentes. Les objets locaux qui n'étaient pas encore contenus dans cette ancienne version du projet ne sont cependant pas effacés de la version locale.

Attribuer un nom à la version

Commande : 'Projet' 'Liaison avec la base de données' 'Attribuer un nom à la version'

Cette commande sert à regrouper l'état actuel des objets sous une seule désignation, ce qui permettra précisément d'appeler cet état ultérieurement. Une boîte de dialogue 'État du projet de <Nom du projet dans la base de données>' s'ouvre. Saisissez une **Étiquette** (label) pour le statut du projet et de manière optionnelle un **commentaire**. Si vous confirmez avec OK, la boîte de dialogue se referme et la désignation ainsi que l'action de l'étiquetage ("désigné par...") apparaissent dans le tableau de l'historique de la version d'un objet individuel aussi bien que de la version du projet. Une version étiquetée ne reçoit pas de numéro de version, mais est caractérisée par une icône d'étiquette dans la colonne 'Version'. Si l'option 'Seulement des noms' est activée, les versions étiquetées sont seules affichées.

Dialogue 'Étiqueter projet <Nom du projet>'

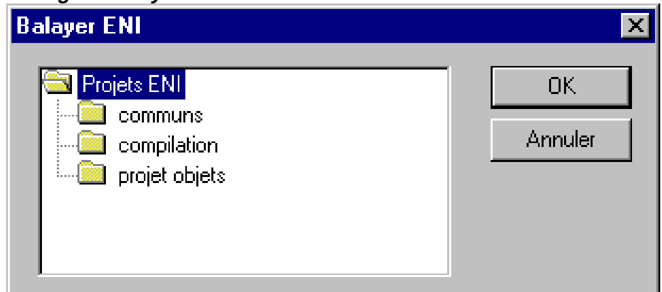


Insérer les objets communs

Commande : 'Projet' 'Liaison avec la base de données' 'Insérer les objets communs'

Cette commande sert à intégrer dans le projet ouvert localement d'autres objets de la catégorie 'Objets communs' disponibles dans la base de données. Pour les objets de la catégorie 'Projet', ceci n'est pas nécessaire puisque lors de la commande 'Dernières versions', tous les objets de la base de données en cours d'utilisation sont automatiquement chargés dans le projet local, même ceux qui n'y avaient pas encore été créés. Pour les objets de la catégorie 'Objets communs' cependant, la commande 'Dernières versions' ne prend en compte que les objets déjà intégrés au projet.

Dialogue 'Balayer ENI'



Insérez un objet supplémentaire comme suit :

La commande ouvre une boîte de dialogue 'Balayer ENI' dans laquelle tous les objets de la base de données du projet sont énumérés, celle-ci étant identifiée par le répertoire du projet indiqué à gauche.

Choisissez la ressource souhaitée et appuyez sur OK ou double-cliquez dessus. Ainsi, l'objet est inséré dans le projet ouvert localement.

Actualiser le statut

Commande : 'Projet' 'Liaison avec la base de données' 'Actualiser le statut'

Cette commande actualise l'affichage dans l'Organisateur d'objets de façon à représenter le statut actuel des objets en rapport à la base de données.

Login

Cette commande ouvre la boîte de dialogue d'ouverture de session par laquelle l'utilisateur doit ouvrir une session au serveur ENI pour chaque catégorie de base de données, de façon à obtenir une connexion du projet vers chaque base de données. Les données d'accès doivent être connues par le serveur ENI (Administration ENI, User Management) et, le cas échéant, par la gestion des utilisateurs de la base de données. Après exécution de la commande, la boîte de dialogue d'ouverture de session pour la catégorie 'Objets de projet' s'ouvre en premier lieu.

Les points suivants y sont affichés :

Host : Adresse du serveur ENI (Hôte), telle qu'elle apparaît également dans Options du projet / catégorie Base de données du projet, dans le champ 'Adresse TCP/IP'.

Projet : Nom du projet dans la base de données (voir également dans les options du projet, catégorie 'Base de données du projet', champ 'Nom du projet')

Introduisez le **Nom de** l'utilisateur et le mot de passe dans la zone **Légitimation**. Si vous souhaitez accéder en tant qu'utilisateur anonyme, laissez le champ 'Nom de' vide.

Appuyez sur OK pour confirmer la saisie. Suite à quoi la boîte de dialogue pour les 'Objets du projet' se referme et la boîte de dialogue d'ouverture de session pour les 'Objets communs' s'ouvre. Saisissez ici également les données correctes d'accès, confirmez avec OK et procédez encore de même pour la troisième boîte de dialogue d'ouverture de session, s'ouvrant pour la catégorie 'Fichiers de compilation'.

La boîte de dialogue d'ouverture de session s'ouvre automatiquement dès que vous tentez d'accéder à une base de données sans vous y avoir auparavant authentifié comme décrit ci-dessus.

Remarque : Si vous souhaitez que les données d'accès à la base de données soient enregistrées avec le projet, activez l'option 'Enregistrer les données d'accès ENI' dans les options de projet, catégorie 'Charger et Enregistrer'.

Dialogue 'Login'

4.4 Gestion des objets

Objet

Sont considérés comme "objets" les modules, les types de données, les visualisations et les ressources (variables acces, variables globales, configuration des variables, histogramme, configuration de l'automate, configuration des tâches, administration d'espion et des recettes). Une partie des dossiers rajoutés pour la structuration du projet sont également inclus. L'ensemble des objets d'un projet se situent dans l'Organisateur d'objets.

Lorsque vous laissez un court instant le pointeur de la souris au-dessus d'un module dans l'Organisateur d'objets, le type de module dont il s'agit (programme, fonction ou bloc fonctionnel) apparaît dans une info-bulle.

Des symboles supplémentaires avant ou après les entrées d’objets caractérisent certains statuts concernant les Fonctions En Ligne et la Connexion ENI vers une base de données.

La fonction Glisser-Déplacer (Drag&Drop) vous permet de déplacer des objets (ainsi que les dossiers, voir 'Dossiers') au sein de leur type d'objet. Pour cela, sélectionnez l'objet et placez-le à l'endroit souhaité en maintenant le bouton gauche de la souris enfoncé. Si une double appellation se produit suite à un déplacement, l’élément qui vient d’être introduit est clairement différencié par l’ajout d’un numéro courant (p.ex. "Objet_1").

Dossier

Pour obtenir une vue d'ensemble de projets de grande envergure, il convient de regrouper les modules, les types de données, les visualisations et les variables globales dans des dossiers.

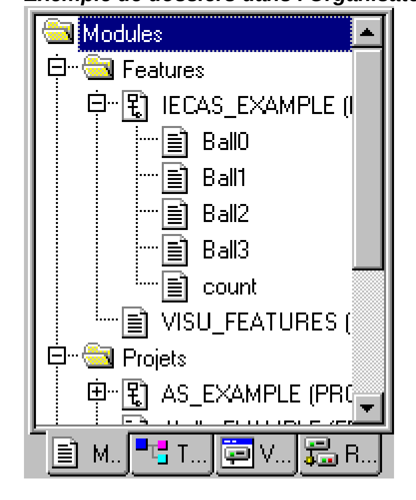
Les dossiers peuvent avoir un nombre illimité de niveaux d'imbrication. Lorsque le symbole du dossier fermé est précédé d'un signe + ⊞, cela signifie que ce dossier contient des objets et/ou d'autres dossiers. En cliquant sur le signe +, le dossier s'ouvre, laissant apparaître les objets subordonnés. En cliquant sur le signe – ⊟ qui apparaît alors, le dossier se ferme. Le menu contextuel propose les commandes 'Développer' et 'Réduire' qui répondent à la même fonctionnalité.

La fonction Glisser-Déplacer (Drag&Drop) vous permet de déplacer les objets ainsi que les dossiers au sein de leur type d'objet. Pour cela, sélectionnez l'objet et placez-le à l'endroit souhaité en maintenant le bouton gauche de la souris enfoncé.

Remarque : Les dossiers n'exercent aucune influence sur le programme. Ils servent simplement à donner un aperçu structuré de votre projet.



Exemple de dossiers dans l'Organisateur d'objets



'Nouveau dossier'

Cette commande permet d'insérer un nouveau dossier en tant qu'objet de rangement. Si un dossier est sélectionné, le nouveau dossier est créé en dessous du dossier sélectionné, sinon il est placé au

même niveau ce dernier. Si une action est sélectionnée, le nouveau dossier est crée au niveau du module auquel l'action appartient.

Pour afficher le menu contextuel de l'Organisateur d'objets qui propose cette commande, il faut sélectionner l'objet ou le type d'objet et appuyer sur le bouton droit de la souris ou effectuer la combinaison de touches <Majuscule>+<F10>.

Le dossier qui vient d'être introduit obtient le nom 'Nouveau dossier'. Veuillez respecter les conventions suivantes en matière de noms de dossiers :

- Les dossiers se trouvant sur le même niveau hiérarchique doivent porter des noms différents. Les dossiers se trouvant à des niveaux différents peuvent porter le même nom.
- Un dossier ne peut avoir le même nom qu'un objet qui se trouve au même niveau.

S'il y a déjà un dossier avec le nom 'Nouveau dossier' sur le même niveau, tout dossier obtenant le même nom se voit attribuer en plus un numéro courant (par exemple „Nouveau dossier 1"). Il est impossible de renommer un dossier d'après un nom déjà existant.

'Développer' 'Réduire'

La commande 'Développer' permet de faire apparaître les objets placés sous l'objet sélectionné, tandis que le commande 'Réduire' permet de les faire disparaître.

Vous pouvez faire apparaître ou disparaître des dossiers en double-cliquant dessus ou en appuyant sur la touche <Entrée>.

Pour afficher le menu contextuel de l'Organisateur d'objets qui propose cette commande, il faut sélectionner l'objet ou le type d'objet et appuyer sur le bouton droit de la souris ou effectuer la combinaison de touches <Majuscule>+<F10>.

'Projet' 'Objet' 'Supprimer un objet'

Raccourci : <Suppr>

Cette commande enlève de l'Organisateur d'objets l'objet actuellement marqué (un module, un type de données, une visualisation ou des variables globales) ou un dossier et son contenu. Cet objet ou ce dossier sont alors supprimés du projet. Avant la suppression effective, il vous est demandé de confirmer votre choix. L'effacement peut être annulé au moyen de la commande 'Éditer' 'Annuler'.

Si la fenêtre de l'éditeur de l'objet est ouverte, elle se ferme automatiquement.

Si l'on utilise la commande 'Editer' 'Couper', pour supprimer l'objet, alors celui-ci est en outre transféré dans le presse-papiers.

'Projet' 'Objet' 'Insérer objet'

Raccourci : <Inser>

Cette commande vous permet de créer un nouvel objet. Le type d'objet (module, type de données, visualisation ou variables globales) dépend de l'onglet choisi dans l'Organisateur d'objets. Veuillez noter à cet égard que l'on utilise un modèle défini le cas échéant pour le type d'objet sélectionné. Ceci est possible pour des objets de type 'Variables globales', 'Type de fichier', 'Fonction', 'Bloc fonctionnel' ou 'Programme', voir ci-dessous, chapitre 'Enregistrer comme modèle'.

Entrez le nom du nouvel objet dans la boîte de dialogue qui s'affiche.

Veuillez noter à cet effet les restrictions suivantes :

- Le nom du module ne peut contenir d'espaces.
- Un module ne peut avoir le même nom qu'un autre module ou qu'un type de donnée.
- Un type de donnée ne peut avoir le même nom qu'un autre type de donnée ou qu'un module.
- Une liste de variables globales ne peut avoir le même nom qu'une autre liste de variables globales.
- Une action ne peut avoir le même nom qu'une autre action au sein du même module.
- Une visualisation ne peut avoir le même nom qu'une autre visualisation.

Dans tous les autres cas, une concordance de nom est permise. Ainsi, des actions provenant de différents modules peuvent avoir le même nom, et une visualisation peut avoir le même nom qu'un module.

S'il s'agit d'un module, vous devez également indiquer le type de module dont il s'agit (programme, fonction ou bloc fonctionnel) et sélectionner le langage dans lequel il doit être programmé. Le **Type de l'unité** est par défaut 'Programme', le **Langage de l'unité de programmation** est celui du dernier module créé. Si un module du type Fonction doit être créé, le type de la valeur renvoyée doit être introduit dans le champ **Type de retour**. À cet effet, tous les types élémentaires de données ou les types définis (tableaux, structures, énumérations, alias) sont autorisés. Vous pouvez utiliser la liste de sélection pour l'édition (p.ex. via <F2>).

Boîte de dialogue Nouveau module



Après confirmation de la saisie via le bouton **OK**, ce qui n'est possible que si vous avez respecté les conventions de noms décrites ci-dessus, le nouvel objet est introduit dans l'Organisateur d'objets et la fenêtre de saisie correspondante est affichée.

Si vous utilisez la commande **'Editer' 'Coller'**, l'objet placé dans le presse-papiers est inséré, sans qu'aucune boîte de dialogue ne s'ouvre. Si le nom de l'objet inséré va à l'encontre des conventions de nom, l'objet sera accompagné d'un caractère de soulignement et d'un numéro courant de manière à le distinguer (p.ex. "Fonction_1").

Si le projet est relié avec une base de données de projet par le biais d'un interface **ENI**, cette connexion peut être configurée de telle sorte que l'on vous demande, lors de la création d'un nouvel objet, dans quelle catégorie de base de données cet objet doit être géré. Dans ce cas, vous obtenez la boîte de dialogue 'Caractéristiques des objets' permettant de sélectionner la catégorie de base de données. Reportez-vous à cet effet à la description des Options de projet pour les bases de données de projet .

'Projet' 'Enregistrer comme fichier modèle'

Les objets de type 'Variables globales', 'Type de fichier', 'Fonction', 'Bloc fonctionnel' ou 'Programme' peuvent être sauvegardés comme modèles de modules. Marquez pour ce faire l'objet dans l'Organisateur d'objets puis sélectionnez la commande 'Enregistrer comme fichier modèle' dans le menu contextuel (bouton droit de la souris). Ensuite, lors de l'Insertion d'un autre objet du même type, la partie de déclaration du modèle sera reprise en premier. On utilise toujours le modèle créé en dernier pour un type d'objet.

'Projet' 'Objet' 'Renommer objet'

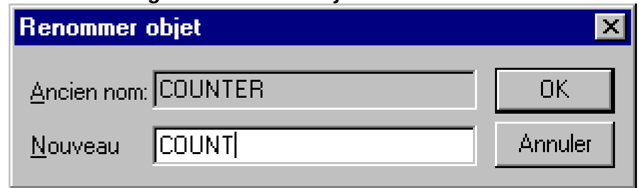
Raccourci : <Barre d'espacement>

Cette commande vous permet de renommer l'objet ou le dossier sélectionné. Veillez à ne pas utiliser un nom existant.

Si la fenêtre de l'éditeur de l'objet est ouverte, le nouveau nom apparaît automatiquement dans la barre de titre.



Boîte de dialogue Renommer objet



'Projet' 'Objet' 'Convertir objet'

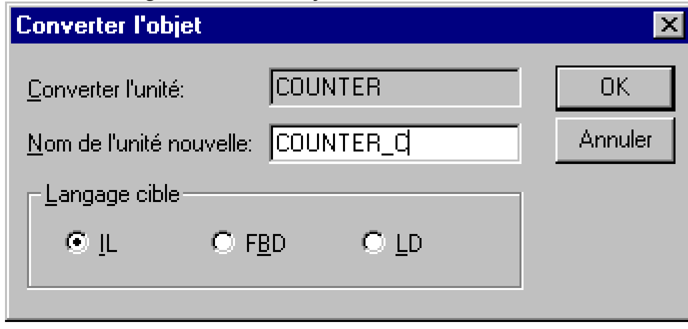
Cette commande n'est exécutable que pour les modules. Vous pouvez compiler des modules créés dans les langages ST, FBD, LD et IL dans l'un des trois langages suivants: IL, FBD, LD.

Pour cela, le projet doit être compilé. Choisissez le langage dans lequel vous souhaitez effectuer la conversion, et donnez un nouveau nom au nouveau module. Veillez à ne pas utiliser un nom existant. Vous pouvez ensuite cliquer sur **OK**. Le nouveau module est ajouté à votre liste de modules.

Le type de traitement effectué lors d'une procédure de conversion correspond à celui d'une compilation.

Attention: Les actions ne peuvent être converties.

Boîte de dialogue Convertir objet

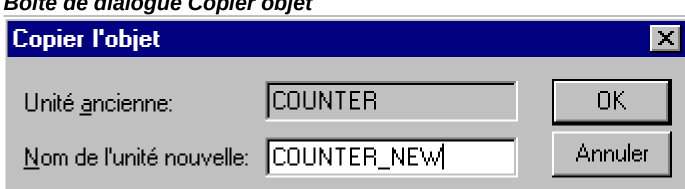


'Projet' 'Objet' 'Copier objet'

Cette commande permet de copier un objet sélectionné et de l'enregistrer sous un nouveau nom. Entrez le nom du nouvel objet ainsi créé dans la boîte de dialogue qui s'affiche. Veillez à ne pas utiliser un nom existant.

En revanche, si vous utilisez la commande 'Editer' 'Copier', l'objet est copié dans le presse-papiers sans qu'aucune boîte de dialogue ne s'ouvre.

Boîte de dialogue Copier objet



'Projet' 'Objet' 'Editer objet'

Raccourci : <Entrée>

Cette commande vous permet de charger, dans l'éditeur pertinent, un objet sélectionné dans l'Organisateur d'objets. Si cet objet est déjà ouvert dans une autre fenêtre, celle-ci passe à l'avant-plan et l'objet peut être édité.

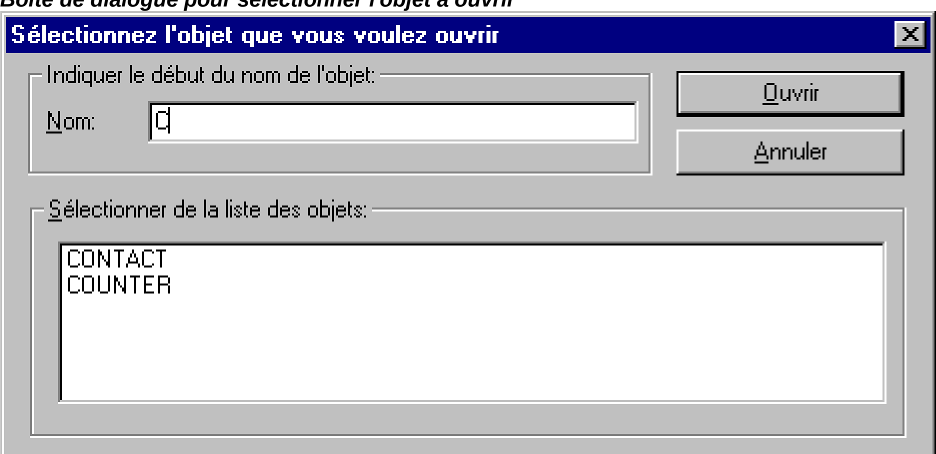
Un objet peut être édité de deux autres manières:

- soit en double-cliquant dessus, soit
- en tapant les premières lettres du nom de l'objet souhaité dans l'Organisateur d'objets. Après quoi, une boite de dialogue s'ouvre et affiche tous les objets du type sélectionné qui commencent par les trois premières lettres introduites dans l'Organisateur d'objets. Sélectionnez l'objet souhaité et

cliquez sur le bouton Ouvrir pour charger l'objet dans la fenêtre de l'éditeur pertinente. Pour le type d'objet Ressources, cette possibilité n'est supportée que pour les variables globales.

Cette dernière possibilité est particulièrement intéressante pour des projets comportant de nombreux objets.

Boîte de dialogue pour sélectionner l'objet à ouvrir



'Projet' 'Objet Propriétés'

Cette commande ouvre la boîte de dialogue 'Propriétés' pour l'objet marqué dans l'Organisateur d'objets.

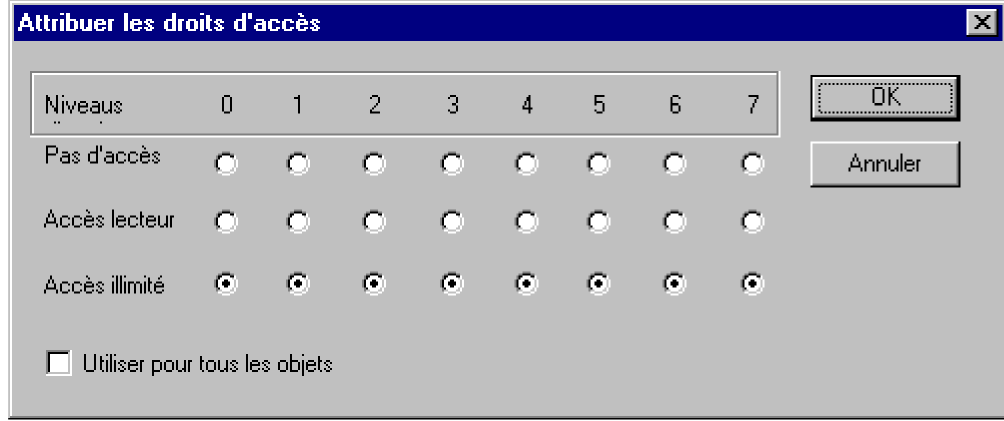
La même boîte de dialogue est accessible via l'onglet 'Droits d'accès' et via la commande 'Projet' 'Objet droits d'accès'' , et on peut s'en servir comme décrit dans cette dernière possibilité.

- Si une liste de variables globales est marquée dans l'Organisateur d'objets, l'onglet 'Liste de variables globales' contient le dialogue 'Liste de variables globales' dans lequel les paramètres pour la mise à jour de la liste et le cas échéant pour l'échange de données de variables globales réseau sont configurés. Les entrées peuvent y être modifiées. Lors de la création d'une liste de Variables globales cette boîte de dialogue est ouverte avec la commande 'Insérer objet', pour autant que le dossier 'Variables globales' ou une des entrées s'y rapportant soit marqué(e) dans l'Organisateur d'objets.
- Si un objet de Visualisation est marqué dans l'Organisateur d'objets et que, dans les configurations du système cible, 'Visualisation de réseau' ou 'Visualisation cible' sont activés, l'onglet 'Visualisation' contient un dialogue dans lequel on doit déterminer si l'objet doit être mis à disposition comme Visualisation Web et/ou comme Visualisation cible .
- Si le projet est relié à une base de données (voir 'Projet' 'Options' 'Base de données du projet'), un autre onglet est à disposition avec le titre 'Liens de base de données'. L'attribution actuelle de l'objet à une des catégories de bases de données ou à la catégorie 'Local' est affichée et vous pouvez la modifier. Vous trouverez de plus amples informations à ce sujet sous 'L'interface ENI CoDeSys .

Objet droits d'accès

Le commande 'Projet' 'Objet' 'Propriétés' permet d'ouvrir la boîte de dialogue d'attribution des droits d'accès aux divers niveaux d'accès: Pour cela, la boîte de dialogue suivante s'affiche:

Boîte de dialogue pour attribuer des droits d'accès



Les membres du niveau 0 peuvent attribuer des droits d'accès individuels à tous les niveaux d'accès. Ces droits d'accès peuvent prendre trois formes:

- **Pas d'accès**: l'objet est inaccessible au niveau d'accès concerné.
- **Lecture seule**: l'objet peut être ouvert en lecture seule par les membres du niveau d'accès; il ne peut donc pas être modifié.
- **Accès complet**: l'objet peut être ouvert et modifié par les membres du niveau d'accès.

Ces configurations s'appliquent soit à l'objet marqué actuellement dans l'Organisateur d'objets, soit à la totalité des modules, types de données, visualisations et ressources du projet. Dans ce dernier cas, vous devez cocher l'option **Etendre à tous les objets**.

L'affectation à un niveau d'accès se fait au moment de l'ouverture du projet par l'intermédiaire d'un mot de passe, dans la mesure où un mot de passe a été attribué par le niveau d'accès 0.

Notez également à ce propos la possibilité supplémentaire d'attribution de droits d'accès relatifs aux groupes de travail et se rapportant à l'utilisation des éléments de visualisation (voir sous Visualisation).

'Projet' 'Ajouter une action'

Cette commande permet de générer une action pour le module sélectionné dans l'Organisateur d'objets. Dans la boîte de dialogue qui s'affiche, vous pouvez sélectionner le nom de l'action et le langage dans lequel celle-ci doit être implémentée.

La nouvelle action est attachée sous son propre module dans l'Organisateur d'objets. Un signe plus apparaît devant le module. Il vous suffit de cliquer sur ce signe pour faire apparaître les objets de l'action. Un signe moins apparaît alors devant le module. En cliquant sur ce signe moins vous masquez les actions et le signe plus apparaît à nouveau. Vous pouvez obtenir le même résultat au moyen des commandes du menu contextuel '**Développer**' et '**Réduire**'.

En double-cliquant sur l'action ou en appuyant sur la touche <Entrée> vous pouvez charger une action dans l'éditeur en vue de l'éditer.

'Projet' 'Ouvrir l'instance'

Cette commande permet d'ouvrir et d'afficher des instances individuelles de blocs fonctionnels en mode En ligne. Avant de procéder à l'ouverture de ces instances, vous devez sélectionner le bloc fonctionnel concerné dans l'Organisateur d'objets. Vous pouvez ensuite sélectionner l'instance souhaitée du bloc fonctionnel ainsi que l’implémentation dans la boîte de dialogue qui s'ouvre. Suite à quoi les éléments souhaités s’affichent dans une fenêtre.

Remarque : Les instances ne peuvent être ouvertes qu'après l'accès au système! (Le projet a été correctement compilé et transféré dans l'automate programmable à l'aide de le commande 'En ligne' 'Accéder au système').

Boîte de dialogue Ouvrir instance

Gestionnaire d'Aide

Instances
Implementation

Ipo.Trafo

OK
Annuler

'Projet' 'Afficher liste de références croisées'

Cette commande donne accès à une boîte de dialogue qui donne un aperçu des endroits où une variable, une adresse ou un module est utilisé. Pour cela, le projet doit avoir été compilé.

Choisissez la **Catégorie Variable**, **Adresse** ou Module, puis entrez le **Nom** de l'élément souhaité. Pour obtenir tous les éléments de la catégorie concernée, introduisez "*" dans le champ Nom.

Un clic sur le bouton **Références croisées** vous donne accès à la liste de tous les endroits où l'élément sélectionné est utilisé. Le nom de la variable (**Variable**) et l'adresse éventuelle (**Adresse**) s'affichent à côté du nom du module et du numéro de ligne ou de réseau. La colonne **Zone** indique s'il s'agit d'une variable locale ou globale, la colonne **Accès** indique si l'accès est en lecture seule ou en écriture. La largeur de la colonne est automatiquement adaptée à l'entrée la plus longue.

Boîte de dialogue Références croisées et exemple de liste des références croisées

Références croisées

Sélection
Catégorie: Variable
Nom: *

Références croisées
Atteindre
Annuler
Dans la fenêtre de message

Liste des références croisées

Module	Variable	Adresse	Zone	Accès
MC_VELOCITYPROFILE (96)	nDirection		Locale	Ecrire
MC_VELOCITYPROFILE (98)	nDirection		Locale	Ecrire
MC_VELOCITYPROFILE (102)	nDirection		Locale	Lire
SMC_MOVEGENERAL (200)	nDirection		Locale	Lire
SMC_MOVEGENERAL (207)	nDirection		Locale	Ecrire
SMC_MOVEGENERAL (209)	nDirection		Locale	Ecrire
SMC_MOVEGENERAL (212)	nDirection		Locale	Lire
DUMMYCONVERTDWORDTO...	fScalefactor		Locale	Lire
DUMMYCONVERTREALTOD...	fScalefactor		Locale	Lire
DUMMYDRIVEINIT (63)	fScalefactor		Locale	Ecrire
MC_READPARAMETER (131)	fScalefactor		Locale	Lire
DUMMYCONVERTDWORDTO...	fFactorVel		Locale	Lire
DUMMYCONVERTREALTOD...	fFactorVel		Locale	Lire
MC_READPARAMETER (133)	fFactorVel		Locale	Lire
DUMMYCONVERTDWORDTO...	fFactorAcc		Locale	Lire

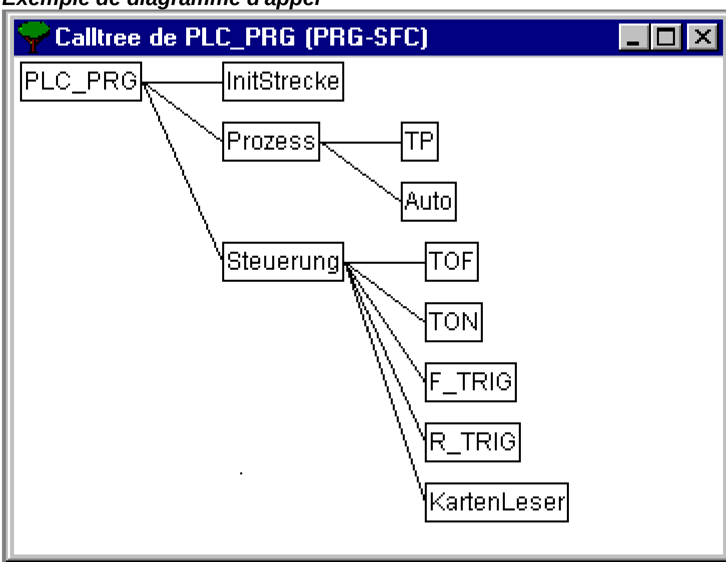
Lorsque vous marquez une ligne dans la liste des références croisées et que vous cliquez sur le bouton **Atteindre** ou lorsque vous double-cliquez sur la ligne concernée, le module s'affiche dans l'éditeur à l'endroit pertinent. Cela vous permet de passer d'un endroit d'utilisation à l'autre sans vous perdre dans des recherches inutiles.

Un moyen de faciliter vos manipulations consiste à placer la liste des références croisées actuelle dans la fenêtre de messages à l'aide du bouton **Dans la fenêtre de messages** de la boîte de dialogue et de passer au module correspondant à partir de la fenêtre de messages.

'Projet' 'Afficher diagramme d'appel'

Cette commande permet d'ouvrir une fenêtre qui représente le diagramme d'appel d'un objet sélectionné dans l'Organisateur d'objets. Pour cela, le projet doit avoir été compilé. Le diagramme d'appel contient aussi bien des appels de modules que des références aux types de données utilisées.

Exemple de diagramme d'appel



4.5 Fonctions d'édition générales

Les commandes décrites ci-dessous sont accessibles dans tous les éditeurs et en partie dans l'Organisateur d'objets. Elles se situent toutes dans le menu 'Éditer' auquel vous accédez par le biais du bouton droit de la souris.

Si le logiciel IntelliPoint est installé sur votre ordinateur, CoDeSys supporte les fonctions de roulette et de bouton de roulette de Microsoft IntelliMouse. Pour tous les éditeurs avec fonction de zoom : Pour agrandir, maintenez la touche <CTRL> enfoncée pendant que vous tournez la roulette en avant. Pour réduire, tournez la roulette en arrière en maintenant la touche <CTRL> enfoncée.

'Editer' 'Annuler'

Raccourci : <Ctrl>+<Z>

Cette commande annule la dernière opération effectuée dans la fenêtre de l'éditeur actuelle ou dans l'Organisateur d'objets, ou, dans le cas d'exécution multiple, elle annule les opérations effectuées jusqu'au moment où la fenêtre a été ouverte. Ceci est valable pour toutes les opérations effectuées dans les éditeurs de modules, de types de données, de visualisation et de variables globales ainsi que dans l'Organisateur d'objets.

La commande 'Editer' 'Refaire' vous permet de rétablir l'opération annulée.

Remarque : Les commandes '**Annuler**' et '**Refaire**' concernent toujours la fenêtre actuelle. Chaque fenêtre dispose de sa propre liste d'opérations. Lorsque vous voulez annuler des opérations effectuées dans plusieurs fenêtres, vous devez activer la fenêtre pertinente. Pour effectuer ces commandes dans l'Organisateur d'objets, celui-ci doit être au premier plan.

'Editer' 'Refaire'

Raccourci : <Ctrl>+<Y>

Cette commande permet de rétablir une action annulée ('Editer' 'Annuler') dans la fenêtre de l'éditeur actuelle ou dans l'Organisateur d'objets. Vous pouvez exécuter autant de fois la commande 'Refaire' que vous avez exécuté la commande 'Annuler'.

Remarque : Les commandes '**Annuler**' et '**Refaire**' concernent toujours la fenêtre actuelle. Chaque fenêtre dispose de sa propre liste d'opérations. Lorsque vous voulez annuler des opérations effectuées dans plusieurs fenêtres, vous devez activer la fenêtre pertinente. Pour effectuer ces commandes dans l'Organisateur d'objets, celui-ci doit être au premier plan.

'Editer' 'Couper'

Icône :

Raccourci : <Ctrl>+<X> ou <Maj>+<Suppr>

Cette commande transfère la sélection actuelle de l'éditeur vers le presse-papiers. L'élément sélectionné est effacé de l'éditeur.

Les objets marqués dans l'Organisateur d'objets n'échappent pas à cette règle, à la différence que certains types d'objets ne peuvent pas être supprimés, p. ex. la configuration de l'automate.

Ne perdez pas de vue que les éditeurs ne supportent pas tous la commande Couper, et que la fonctionnalité de cette commande peut être limitée dans certains éditeurs.

La forme de la sélection dépend de l'éditeur concerné:

Dans les éditeurs de texte (IL, ST, déclarations), la sélection prend la forme d'une liste de caractères.

Dans les éditeurs FBD et LD, la sélection prend la forme d'un ensemble de réseaux, chacun d'eux étant marqué par un rectangle en pointillés dans les champs numériques, ou d'une zone comportant toutes les lignes, zones et opérandes qui la précèdent.

Dans l'éditeur SFC, la sélection est une partie de la série d'étapes, entourée d'un rectangle en pointillés.

Pour insérer le contenu du presse-papiers, utilisez la commande 'Editer' 'Coller'. Dans l'éditeur SFC, vous pouvez également utiliser les commandes 'Extras' 'Insérer séquence parallèle (droite)' ou 'Extras' 'Insérer derrière'.

Pour insérer une sélection dans le presse-papiers sans l'effacer, utilisez la commande 'Editer' 'Copier'.

Pour effacer une zone marquée sans modifier le contenu du presse-papiers, utilisez la commande 'Editer' 'Supprimer'.

'Editer' 'Copier'

Icône : Raccourci : <Ctrl>+<C>

La forme de la sélection dépend de l'éditeur concerné:

Dans les éditeurs de texte (IL, ST, déclarations), la sélection prend la forme d'une liste de caractères.

Dans les éditeurs FBD et LD, la sélection prend la forme d'un ensemble de réseaux, chacun d'eux étant marqué par un rectangle en pointillés dans les champs numériques, ou d'une zone comportant toutes les lignes, zones et opérandes qui la précèdent.

Dans l'éditeur SFC, la sélection est une partie de la série d'étapes, entourée d'un rectangle en pointillés.

Pour insérer le contenu du presse-papiers, utilisez la commande 'Editer' 'Coller'. Dans l'éditeur SFC, vous pouvez également utiliser les commandes 'Extras' 'Insérer séquence simultanée (droite)' ou 'Extras' 'Insérer derrière'.

Pour effacer une zone marquée et la déplacer en même temps dans le presse-papiers, utilisez la commande 'Editer' 'Couper'.

'Editer' 'Coller'

Icône : **Raccourci : <Ctrl>+<V>**

Cette commande insère le contenu du presse-papiers à la position actuelle dans la fenêtre de l'éditeur. Dans les éditeurs graphiques, cette commande n'est exécutable que lorsque l'insertion se fait dans le respect de la structure.

Pour ce qui est de l'Organisateur d'objets, l'objet est inséré à partir du presse-papiers.

Ne perdez pas de vue que les éditeurs ne supportent pas tous la commande Coller, et que la fonctionnalité de cette commande peut être limitée dans certains éditeurs.

La position actuelle varie d'un type d'éditeur à l'autre:

Dans les éditeurs de texte (IL, ST, déclarations), la position actuelle correspond à celle du curseur clignotant (une petite ligne verticale que l'on positionne à l'aide de la souris).

Dans les éditeurs FBD et LD, la position actuelle est celle du premier réseau doté d'un rectangle en pointillés dans la zone numérique du réseau. Le contenu du presse-papiers est inséré devant ce réseau. Si l'élément copié est une structure partielle, celle-ci est insérée devant l'élément marqué.

Dans l'éditeur SFC, la position actuelle est déterminée par la zone sélectionnée entourée d'un rectangle en pointillés. Le contenu du presse-papiers est inséré devant la sélection ou dans une nouvelle branche (simultanée ou alternative) à gauche de l'élément sélectionné. Cela dépend de la sélection et du contenu du presse-papiers.

Dans l'éditeur SFC, vous pouvez également insérer le contenu du presse-papiers à l'aide des commandes 'Extras' 'Insérer séquence simultanée (droite)' ou 'Extras' 'Insérer derrière'.

Pour insérer une sélection dans le presse-papiers sans l'effacer, utilisez la commande 'Editer' 'Copier'.

Pour effacer une zone marquée sans modifier le contenu du presse-papiers, utilisez la commande 'Editer' 'Supprimer'.

'Editer' 'Supprimer'

Raccourci : <Suppr>

Cette commande supprime la zone marquée de la fenêtre de l'éditeur. Le contenu du presse-papiers reste inchangé.

Les objets marqués dans l'Organisateur d'objets n'échappent pas à cette règle, à la différence que certains types d'objets ne peuvent pas être supprimés, p. ex. la configuration de l'automate.

La forme de la sélection dépend de l'éditeur concerné:

Dans les éditeurs de texte (IL, ST, déclarations), la sélection prend la forme d'une liste de caractères.

Dans les éditeurs FBD et LD, la sélection prend la forme d'un ensemble de réseaux, chacun d'eux étant marqué par un rectangle en pointillés dans les champs numériques.

Dans l'éditeur SFC, la sélection est une partie de la série d'étapes, entourée d'un rectangle en pointillés.

Dans la gestionnaire de bibliothèques, la sélection correspond au nom de bibliothèque actuellement sélectionné.

Pour effacer une zone marquée et la déplacer en même temps dans le presse-papiers, utilisez la commande 'Editer' 'Couper'.

'Editer' 'Rechercher'


Icône :

Cette commande vous permet de rechercher un texte spécifique dans la fenêtre de l'éditeur actuelle. Elle ouvre la boîte de dialogue Rechercher. Celle-ci reste ouverte jusqu'à ce que vous cliquiez sur le bouton **Annuler**.

Vous pouvez saisir le chaîne de caractères recherchée dans le champ **Rechercher**.

En outre, vous pouvez effectuer la recherche sur un **Mot entier** ou sur une partie de mot, vous pouvez cibler la recherche en fonction de la casse (option **Respecter la casse)** et définir le sens de la recherche (**vers le haut** ou **vers le bas**). Veuillez noter que dans le cas des modules FBD, le traitement s'effectue de droite à gauche !

Le bouton **Nouvelle recherche** lance la recherche. Celle-ci commence à l'endroit que vous avez choisi et elle est effectuée dans le sens que vous avez spécifié. Lorsque le texte recherché est trouvé, il apparaît marqué. Dans le cas contraire, un message vous le signale. Des recherches successives peuvent être effectuées jusqu'à ce que le début ou la fin de la fenêtre de l'éditeur est atteint.

Ne perdez pas de vue que le texte trouvé peut être masqué par la boîte de dialogue **Rechercher**.

Boîte de dialogue Rechercher

'Editer' 'Rechercher le suivant'

Icône : Raccourci : <F3>

Cette commande vous permet d'effectuer une recherche selon les mêmes paramètres que ceux de la commande 'Editer' 'Rechercher'**.** Veuillez noter que dans le cas des modules FBD, le traitement s'effectue de droite à gauche !

'Editer' 'Remplacer'

Cette commande vous permet de rechercher un texte spécifique, exactement comme la commande 'Editer' 'Rechercher', et de le remplacer par un autre. Après avoir choisi cet ordre, la boîte de dialogue Rechercher et remplacer s'ouvre; elle reste ouverte tant que vous ne cliquez pas sur le bouton **Annuler** ou **Fermer**.

Le bouton **Remplacer** remplace la sélection actuelle par le texte saisi dans le champ **Remplacer par** de la boîte de dialogue.

Le bouton **Remplacer tout** substitue toutes les occurrences du texte saisi dans le champ **Rechercher**, à partir de la position actuelle, par le texte saisi dans le champ **Remplacer par**. Après cette opération, un message vous indique le nombre de substitutions opérées.

Boîte de dialogue Rechercher et Remplacer

'Editer' 'Liste de sélection pour l'édition'

Raccourci : <F2>

Cette commande donne accès à une boîte de dialogue proposant de choisir parmi les entrées possibles compte tenu de la position actuelle du curseur. Dans la colonne de gauche de la boîte de dialogue, choisissez la catégorie d'entrée souhaitée, puis marquez l'entrée souhaitée dans la colonne de droite et confirmez votre choix en cliquant sur **OK**. Votre sélection est insérée à la position actuelle du curseur dans la fenêtre de l'éditeur.

Les catégories proposées dépendent de la position actuelle du curseur dans la fenêtre de l'éditeur; en d'autres termes, elles dépendent des entrées qu'il est possible d'effectuer à cet endroit (p. ex. variables, opérateurs, modules, conversions, etc.).

Si l'option **Avec des arguments** est activée, les arguments à passer sont indiqués lors de l'insertion de l'élément sélectionné. Exemples : Sélection du bloc fonctionnel fu1, qui a défini la variable d'entrée var_in: fu1(var_in:=);

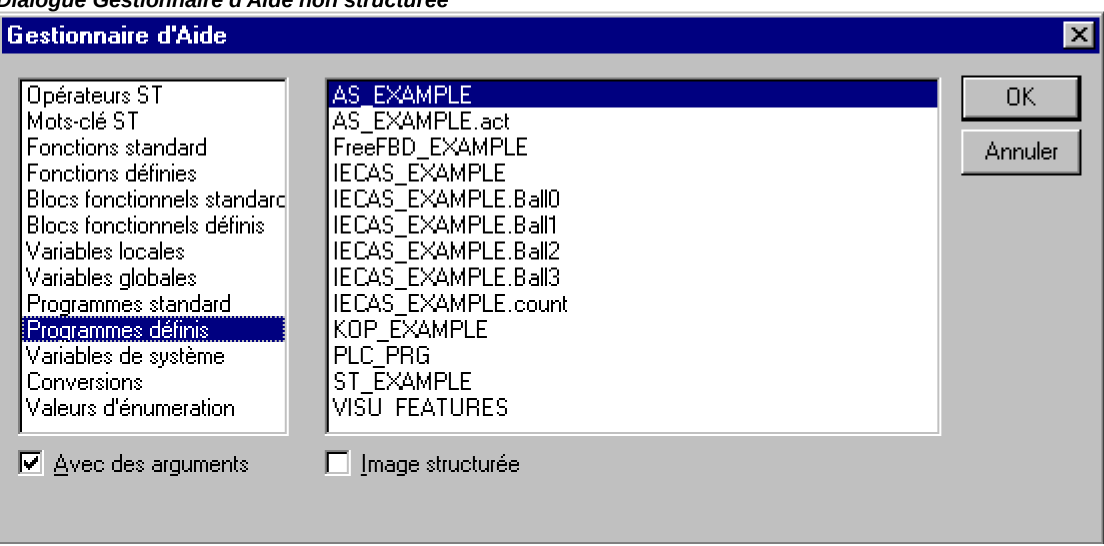
Insertion de la fonction func1, qui nécessite les paramètres var1 et var2 : func1(var1,var2);

En principe, il est possible d'alterner entre une représentation structurée ou non des éléments à disposition. Ceci se produit en activant ou en désactivant l'option **Image structuré.**

Veuillez noter : Lors de l'introduction de variables, vous avez en outre la possibilité d'utiliser la fonction 'Intellisense' comme aide.

Image non structurée

Dialogue Gestionnaire d'Aide non structurée



Les modules, variables ou types de fichiers de chaque catégorie sont tout simplement classés par ordre alphabétique et de façon linéaire.

Pour certaines positions (p. ex. dans la liste d'espion), il peut arriver que des noms composés de variables soient nécessaires. La boîte de dialogue Liste de sélection pour l'édition affiche alors une liste de tous les modules, et les variables globales sont représentées par un point. Le nom de chaque module est suivi d'un point. Si vous marquez un module par un double-clic ou en appuyant sur la touche <Entrée>, vous ouvrez la liste des variables s'y rapportant. Si des instances et types de données sont disponibles, ils peuvent être affichés. Pour prendre en compte la variable sélectionnée en fin de compte, cliquez sur **OK**.

Image structurée

Si **Image structurée** est sélectionné, les modules, variables ou types de données sont classés de manière hiérarchique. Ceci est possibles pour Programmes standard, Fonctions standard, Blocs fonctionnels standard, Programmes définis, Fonctions définies, Blocs fonctionnels définis, Variables globales, Variables locales, Types définis, Types, Variables d'espion. L'apparence et la hiérarchie correspondent à celles de l'Organisateur d'objets, et si des éléments provenant de bibliothèques sont concernés, ceux-ci sont insérés à la position supérieure et par ordre alphabétique, et la hiérarchie est représentée comme dans le gestionnaire de bibliothèques.

Les variables d'entrée et de sortie des blocs fonctionnels, déclarées comme variables locales ou globales, sont affichées dans la catégorie 'Variables locales' ou 'Variables globales' en dessous du nom de l'instance (p.ex. Inst_TP.ET, Inst_TP.IN, ...). On y accède en sélectionnant le nom de l'instance (p.ex. Inst_TP) et en confirmant avec **OK**.

Dialogue Gestionnaire d'Aide

Gestionnaire d'Aide

Opérateurs ST
Mots-clé ST
Fonctions standard
Fonctions définies
Blocs fonctionnels standard
Blocs fonctionnels définis
Variables locales
Variables globales
Programmes standard
Programmes définis
Variables de système
Conversions
Valeurs d'énumeration

Programmes définis
Features
Languages
AS_EXAMPLE (PRG)
FreeFBD_EXAMPLE (PRG)
KOP_EXAMPLE (PRG)
ST_EXAMPLE (PRG)
PLC_PRG (PRG)

OK
Annuler

Avec des arguments
Image structurée

Si l'instance d'un bloc fonctionnel est marquée, l'option **Avec des arguments** peut être sélectionnée. Le nom de l'instance et les paramètres d'entrée du bloc fonctionnel seront alors insérés pour les langages littéraux IL et ST ainsi que dans la configuration des tâches.

Par exemple, pour la sélection Inst (Déclaration Inst: TON;),

Inst(IN:= ,PT:=) est inséré.

Si l'option n'est pas sélectionnée, le nom d'instance est seul inséré. Avec les langages graphiques ou encore dans la fenêtre d'espion, seul le nom d'instance est généralement inséré.

Les composants de structures sont représentés de la même façon que les instances de blocs fonctionnels.

Pour les énumérations, les valeurs individuelles d'énumération sont reprises sous le type d'énumération. L'ordre est : Énumérations provenant de bibliothèques, énumérations provenant de types de données, énumérations locales provenant de modules.

En règle générale, les lignes qui contiennent des sous-objets ne peuvent être sélectionnées (à l'exception des instances, voir ci-dessus), mais peuvent être masquées ou affichées de la même façon que les noms composés de variables.

Si vous appelez la liste de sélection pour l'édition dans le gestionnaire d'espion et de recettes ou lors du choix des variables d'histogramme dans la boîte de dialogue de configuration de l'histogramme, vous pouvez procéder à un choix multiple. En maintenant la touche <Maj> enfoncée, vous pouvez sélectionner un bloc de variables; la touche <Ctrl> permet de sélectionner plusieurs variables séparément. Les variables sélectionnées sont marquées. Si, lors du marquage du bloc, vous sélectionnez des lignes ne contenant pas de variables valables (p.ex. des noms de modules), celles-ci ne seront pas reprises dans la sélection. Il n'est pas possible de marquer de telles lignes lors d'une sélection individuelle.

Dans la fenêtre d'espion ou dans la configuration de l'histogramme, vous pouvez reprendre des structures, tableaux ou instances hors de la liste de sélection pour l'édition. Vu que l'affichage ou le masquage des éléments est défini par un double-clic de la souris, la sélection ne peut s'effectuer dans ces cas que par **OK**.

Suite à quoi les variables sélectionnées sont introduites ligne par ligne dans la fenêtre d'espion, chaque variable sélectionnée étant donc écrite dans une ligne. Pour les variables de l'histogramme, chacune d'entre elle est introduite dans une ligne dans la liste des variables de l'histogramme.

Si vous dépassez le nombre maximum de 20 variables d'histogramme lors de l'introduction des variables sélectionnées, le message d'erreur suivant apparaît : "Vous ne pouvez introduire plus de 20 variables". Toute variable sélectionnée au-delà de ces vingt n'est pas reprise dans la liste.

Remarque: Pour certaines entrées (p. ex. les variables globales), la mise à jour est effectuée après une compilation.

'Editer' 'Déclarer automatiquement'

Raccourci : <Maj>+<F2>

Cette commande vous permet d'obtenir une boîte de dialogue pour déclarer des variables dès que vous entrez une nouvelle variable dans l'éditeur de déclaration. Cette boîte de dialogue s'ouvre également lorsque l'option de projet 'Déclarer automatiquement' activée.

'Editer' 'Erreur prochaine'

Raccourci : <F4>

Lorsqu'un projet n'a pas pu être compilé à cause de la présence d'erreurs, cette commande permet d'afficher la prochaine erreur. La fenêtre de l'éditeur pertinente est activée, l'emplacement contenant l'erreur est marqué et le message d'erreur correspondant est affiché simultanément dans la fenêtre de messages. Si ces avertissements devaient être ignorés en travaillant pas à pas avec F4, l'option 'F4 ignore les avertissements' doit être activée dans le menu 'Projet' 'Options' 'Environnement de travail'.

'Editer' 'Erreur précédente'

Raccourci : <Maj>+<F4>

Lorsqu'un projet n'a pas pu être compilé à cause de la présence d'erreurs, cette commande permet d'afficher l'erreur précédente. La fenêtre de l'éditeur pertinente est activée, l'emplacement contenant l'erreur est marqué et le message d'erreur correspondant est affiché simultanément dans la fenêtre de messages. Si ces avertissements devaient être ignorés en travaillant pas à pas avec F4, l'option 'F4 ignore les avertissements' doit être activée dans le menu 'Projet' 'Options' 'Environnement de travail'.

'Editer' 'Macros'

Toutes les macros définies pour le projet en cours apparaissent sous cette option. (En ce qui concerne la création, reportez-vous à 'Projet' 'Options' 'Macros'). Si vous sélectionnez une macro et qu'elle est exécutable, la boîte de dialogue 'Exécuter la macro' s'ouvre. Le nom de la macro s'affiche ainsi que la ligne de commande en cours. Le bouton **Annuler** permet d'interrompre l'exécution de la macro, tout en terminant l'exécution de la commande en cours. Un message correspondant apparaît dans la fenêtre des messages et s'ajoute au journal dans le mode En ligne : '<Macro>: exécution interrompue par l'utilisateur'.

Vous pouvez procéder à l'exécution des macros En ligne et Hors ligne. Les commandes se rapportant au mode concerné seront seules exécutées.

4.6 Fonctions en ligne générales

Les commandes en ligne qui sont à votre disposition sont groupées dans le menu '**En ligne**'. L'exécution de certains de ces commandes dépend de l'éditeur actif.

L'accès à ces commandes passe par l'accès au système.

La fonctionnalité '**En Ligne**' vous permet de procéder à des changements du programme sur l'automate en cours de fonctionnement. Voyez à cet effet 'En Ligne 'Accéder au système'.

Les sections suivantes décrivent chaque commande En ligne à titre individuel :

'En Ligne' 'Accéder au système'

Icône : **Raccourci : <Alt>+<F8>**

Cette commande établit la liaison entre le système de programmation et l'automate programmable (ou lance le programme de simulation) et bascule en mode En Ligne.

Si le projet actuel n'a pas été compilé depuis qu'il a été ouvert ou modifié pour la dernière fois, la compilation est lancée (voir 'Projet' - 'Compiler'). Si la compilation se heurte à une erreur, CoDeSys ne bascule pas en mode En Ligne.

Si le projet en cours a été modifié mais pas fermé depuis le dernier téléchargement sur l'automate, et si les informations relatives à ce dernier téléchargement n'ont pas été effacées par la commande 'Projet' 'Réorganiser tout', une boîte de dialogue s'ouvre après la commande 'Accéder au système', vous demandant : "Le programme a été modifié! Voulez-vous charger les modifications ? (**Online Change**)". (Voyez à ce sujet les remarques relatives aux changements En ligne ci-dessous). Si vous appuyez sur **Oui**, vous confirmez que les parties modifiées du projet doivent être chargées sur l'automate lors de l'ouverture de session. **Non** permet une ouverture de session sans que les changements effectués depuis le dernier téléchargement soient chargés sur l'automate. **Annuler** vous permet d'interrompre la commande. **Charger tout** permet de charger le projet complet sur l'automate.

Dialogue d'ouverture de session étendu

CoDeSys

Le programme a été modifié! Voulez-vous lancez le nouveau programme?

Oui | Non | Annuler | Détails<<

Projet dans l'espace de travail		Projekt dans l'automate	
Nom du	Bitoffset.pro	Nom du	Bitoffset.pro
Modifié le:	12.2.04 14:28:14	Modifié le:	12.2.04 12:35:46
Titre:	proj_tempcontr	Titre:	proj_tempcontr
Auteur:	XY	Auteur:	XY
Version:	1.8	Version:	1.5
Description:	Temperature Control 1	Description:	Temperature Control 1

Lorsque l'option 'Mode En ligne en mode de sécurité' a été activée dans les options de projet, catégorie Environnement de travail, et si le système cible supporte cette fonction, ce dialogue peut être étendu par le biais du bouton **Détails** Les **informations relatives au projet** actuellement chargé dans le système de programmation et au projet déjà disponible au niveau de l'automate sont alors affichées en plus.

Veuillez noter que le bouton par défaut, c'est-à-dire celui sur lequel l'accent est automatiquement mis, dépend de la configuration du système cible.

Si l'accès au système n'a rencontré aucun problème, toutes les fonctions en ligne sont à votre disposition (dans la mesure où la catégorie Options de compilation de la boîte de dialogue '**Options**' a été configurée dans ce sens). Pour toutes les déclarations de variables visibles à l'écran, les valeurs actuelles sont espionnées.

Pour basculer du mode en ligne au mode hors ligne, utilisez la commande 'Quitter le système' du menu 'En Ligne'**.**

Remarques relatives aux changements En ligne:

- Un changement En ligne n'est pas possible après des modifications au niveau de la configuration des tâches ou de la configuration de l'automate, après l'insertion d'une bibliothèque ou suite à la commande 'Projet' 'Réorganiser tout' (voir plus bas).
- Lorsque les informations de téléchargement (fichier <Nom de projet><Targetidentifier>.ri), créées lors du dernier chargement du projet, ont été effacées (par exemple par le biais de la commande 'Réorganiser tout'), aucun changement En ligne n'est possible, à moins que le fichier *.ri n'ait été enregistré en un autre emplacement sous un autre nom et qu'il puisse à nouveau être chargé par le biais de la commande 'Charger les informations de téléchargement'. Voyez à cet effet le point 'Changement En ligne pour un projet....'.
- Lors d'un changement En ligne, il n'est pas procédé à une nouvelle initialisation, ce qui signifie que les changements des valeurs d'initialisation ne sont pas pris en compte !
- Les variables rémanentes gardent leurs valeurs lors d'un changement En ligne, contrairement à un nouveau téléchargement du projet (voir ci-dessous, 'En ligne' 'Charger').

Changement En ligne pour un projet fonctionnant sur plusieurs automates :

Si vous utilisez un projet proj.pro sur deux automates programmables identiques API1 et API2 (système cible identique), et si vous souhaitez, après avoir modifié le projet, pouvoir actualiser ce même projet sur les deux automates par le biais des fonctions En ligne, veuillez procéder comme suit:

- Établissez une connexion avec l'automate API1 (En ligne/Paramètres de communication) puis **chargez *proj.pro* sur API1** (En ligne/Accéder au système, Charger). En parallèle au *proj.pro*, le fichier *proj00000001.ri* est alors crée, ce dernier contenant les informations de téléchargement.
- **Renommez *proj00000001.ri*** , p.ex. en *proj00000001_API1.ri*. Cette "sauvegarde" sous un fichier d'un autre nom est nécessaire car lors d'un téléchargement ultérieur de *proj.pro*, le fichier *proj00000001.ri* contenant les nouvelles informations de téléchargement serait écrasé, ce qui provoquerait la perte des informations relatives au téléchargement sur API1.
  Démarrez le projet et quittez à nouveau le système.
- Établissez maintenant la connexion avec l'automate API2 puis **chargez *proj.pro* sur API2**. Ensuite, en parallèle au *proj.pro*, un fichier *proj00000001.ri* est à nouveau crée, ce dernier contenant les informations du téléchargement actuel.
- Sauvegardez ce nouveau fichier *proj00000001.ri* en le **renommant** p.ex. *proj00000001_API2.ri*.
- Démarrez le projet sur API2 et quittez à nouveau le système.
- Effectuez maintenant dans **CoDeSys** les modifications de *proj.pro* que vous souhaitez charger dans le projet fonctionnant sur l'automate par le biais des Changements En ligne.
- Pour pouvoir exécuter le **changement En ligne à *proj.pro* sur API1**, les informations de téléchargement de proj.pro sauvegardées sur API1 doivent tout d'abord être rendus à nouveau disponibles. Pour ce faire, CoDeSys recherche le fichier *proj00000001.ri* lors de l'ouverture de session. Vous avez sauvegardé le fichier .ri valide pour API1 sous *proj00000001_API1.ri*.
  Vous avez alors deux possibilités :
  (a) Vous pouvez à nouveau renommer le fichier proj00000001_API1.ri en *proj00000001.ri*. Ainsi, vous êtes assuré que lors d'une ouverture de session sur API1, les informations correctes de téléchargement seront prises en compte automatiquement et le changement en ligne sera proposé.
  (b) Vous pouvez également charger de manière ciblée le fichier *proj00000001_API1.ri* par le biais de la commande 'Projet' 'Charger informations de téléchargement', cela avant même d'ouvrir une session. Vous évitez ainsi de devoir renommer le fichier.

- Afin de pouvoir également effectuer le **changement en ligne de *proj.pro* sur API2**, veuillez procéder comme décrit au point 8, avec bien entendu le fichier *proj00000001_API2.ri*.

Si le système affiche

Message d'erreur :

```
"Le profil de l'automate sélectionné ne correspond pas à celui du système cible!
La connexion sera terminée."
```

Vérifiez si le système cible tel qu'il a été paramétré dans la configuration du système cible (Ressources) concorde avec le paramétrage 'En Ligne' **'Paramètres de communication'.**

Message d'erreur :

```
"Faute de communication: Logout."
```

Vérifiez si l'automate fonctionne. Vérifiez si les paramètres configurés dans `En Ligne' `Paramètres de communication' concordent avec les paramètres de votre automate programmable. Vous devez vérifier en particulier si le port adéquat a été réglé et si les débits en bauds concordent sur l'automate et dans le système de programmation. Si vous utilisez la gateway, vérifiez si le bon canal est réglé.

Message d'erreur :

```
"Le programme a été modifié! Voulez-vous charger les modifications ?"
```

Le projet en cours d'utilisation dans l'éditeur ne correspond pas au programme chargé actuellement dans l'automate. Il est par conséquent impossible d'effectuer l'espionnage et le débogage. Vous pouvez répondre **Non**, quitter le système et ouvrir le projet pertinent, ou répondre **Oui** et charger le projet actuel dans l'automate programmable.

Message :

```
"Le programme a été modifié! Voulez-vous charger les modifications ? (Online Change)"
```

Le projet est en cours d'utilisation sur l'automate. Le système cible supporte les 'Changements En ligne' et le projet a été modifié par rapport au dernier téléchargement ou par rapport au dernier Changement En ligne. Vous pouvez déterminer si ces modifications doivent être chargées sur l'automate, le programme étant en cours, ou si la commande doit être interrompue. Vous pouvez également charger le code compilé entier en appuyant sur le bouton **Charger tout**.

'En Ligne' 'Quitter le système'

Icône : **Raccourci : <Ctrl>+<F8>**

Cette commande coupe la liaison avec l'automate programmable ou quitte le programme de simulation et vous fait basculer en mode hors ligne.

Pour basculer à nouveau en mode en ligne, utilisez le commande 'Accéder au système' du menu 'En ligne'.

'En Ligne' 'Lancer'

Cette commande charge le projet compilé dans l'automate programmable.

Si vous utilisez la génération de code C, le compilateur C qui génère le fichier de téléchargement est appelé avant que ne soit chargé le projet dans l'automate programmable. Sinon, ce fichier est généré dès la compilation.

Les informations relatives au téléchargement sont sauvegardées dans un fichier <Nom du projet>0000000ar.ri qui sera utilisé lors des changements En ligne afin de comparer le programme en cours avec celui chargé en dernier lieu sur l'automate, si bien que les parties modifiées seront seules chargées à nouveau. Vous pouvez effacer ce fichier à l'aide de la commande 'Projet' 'Réorganiser tout'. Voir pour Changement En ligne d'un projet sur plusieurs automates au chapitre 4.6, ''En Ligne' 'Accéder au système'.

En fonction du système cible, il est possible de créer automatiquement un nouveau fichier *.ri en mode Hors ligne lors de la création d'un projet d'initialisation.

'En Ligne' 'Démarrer'

Icône : **Raccourci : <F5>**

Cette commande lance l'exécution du programme utilisateur dans l'automate programmable ou dans la simulation.

Vous pouvez choisir d'exécuter cette commande directement après l'exécution de la commande 'En ligne' 'Lancer' ou après avoir arrêté l'exécution du programme utilisateur dans l'automate

programmable à l'aide de la commande 'En ligne' 'Arrêter' ou encore lorsque le programme utilisateur est à un point d'arrêt ou lorsque qu'un cycle individuel (commande 'En Ligne' 'Cycle indépendant') a été exécuté.

'En Ligne' 'Arrêter'

Icône :  **Raccourci : <Maj>+<F8>**

Cette commande suspend, entre deux cycles, l'exécution du programme utilisateur dans l'automate programmable ou dans la simulation.

Pour relancer le programme utilisateur, utilisez la commande 'En ligne' 'Démarrer'.

'En Ligne' 'Reset'

Cette commande ramène toutes les variables (à l'exception des variables rémanentes (VAR RETAIN)) à la valeur qu'elles avaient lors de leur initialisation (y compris donc les variables déclarées à l'aide de VAR PERSISTENT!). Les variables qui ne sont pas explicitement munies d'une valeur d'initialisation, sont ramenées à la valeur standard d'initialisation (par exemple, les nombres entiers sont mis à zéro). Avant l'écrasement de toutes les variables, CoDeSys demande confirmation. C'est la même situation que lors d'une panne de courant ou lors d'une mise en/hors service de l'automate (démarrage à chaud) durant le fonctionnement du programme.

Pour relancer le programme, utilisez la commande 'En ligne' 'Démarrer'.

Voir également à cet effet 'En Ligne' 'Reset (origine)' et 'En Ligne' 'Reset (à froid)'.

'En Ligne' 'Reset (à froid)'

Cette commande ramène toutes les variables, y compris les variables rémanentes, à la valeur qu'elles avaient lors de leur initialisation. C'est la même situation que lors du démarrage d'un programme qui a été chargé pour la première fois dans l'automate (démarrage à froid). Voir également à cet effet 'En Ligne' 'Reset' et 'En Ligne' 'Reset (origine)'.

'En Ligne' 'Reset origine'

Cette commande ramène toutes les variables, y compris les variables rémanentes (VAR RETAIN et VAR PERSISTENT), à la valeur qu'elles avaient lors de leur initialisation et efface le programme utilisateur sur l'automate. L'automate est ramené à son état original. Voir également à cet effet 'En Ligne' 'Reset' et 'En Ligne' 'Reset (à froid)'.

'En Ligne' 'Point d'arrêt actif/inactif'

Icône : Raccourci : <F9>

Cette commande définit un point d'arrêt à la position actuelle dans la fenêtre active. Si un point d'arrêt y est déjà défini, il est enlevé.

La position où un point d'arrêt peut être défini dépend du langage dans lequel est écrit le module dans la fenêtre active.

Dans les éditeurs de texte (IL, ST) le point d'arrêt est défini sur la ligne où se trouve le curseur, lorsque cette ligne est une position de point d'arrêt (on peut le déterminer grâce à la couleur gris foncé du champ de numérotation de ligne). Pour définir ou enlever un point d'arrêt dans les éditeurs de texte, vous pouvez cliquer sur le champ de numérotation de ligne.

Dans les éditeurs FBD et LD, le point d'arrêt est placé sur le réseau actuellement marqué. Pour définir ou enlever un point d'arrêt dans ce type d'éditeur, vous pouvez cliquer sur le champ de numérotation de réseau.

Dans l'éditeur SFC, le point d'arrêt est défini sur l'étape marquée actuellement. Pour définir ou enlever un point d'arrêt dans l'éditeur SFC, vous avez la possibilité d'utiliser la touche <Maj> associée à un double-clic de la souris.

Lorsqu'un point d'arrêt est défini, le champ de numérotation de ligne, de réseau ou l'étape apparaissent dans une couleur de fond bleu clair.

Lorsque le programme atteint un point d'arrêt en cours d'exécution, il s'arrête et la couleur de fond du champ correspondant apparaît en rouge. Pour poursuivre, utilisez les commandes 'En ligne' 'Démarrer' ou 'En Ligne' 'Etape individuelle dans' ou 'En Ligne' 'Etape individuelle sur'.

'En Ligne' 'Dialogue des points d'arrêt'

Cette commande ouvre une boîte de dialogue permettant d'éditer les points d'arrêt de l'ensemble du projet. En outre, cette boîte de dialogue affiche l'ensemble des points d'arrêt actuellement définis.

Pour définir un point d'arrêt, sélectionnez un module dans la zone de liste modifiable pertinente de la boîte de dialogue et, dans la zone de liste modifiable **Emplacement**, choisissez la ligne ou le réseau où vous souhaitez définir un point d'arrêt et cliquez sur le bouton **Ajouter**. Le point d'arrêt est intégré à la liste.

Pour supprimer un point d'arrêt, sélectionnez-le dans la liste et cliquez sur le bouton **Effacer.**

Le bouton **Effacer tout** supprime tous les points d'arrêt.

Pour vous rendre à un endroit de l'éditeur où un point d'arrêt a été défini, marquez ce point d'arrêt dans la liste des points d'arrêt définis et cliquez sur le bouton **Atteindre.**

Pour définir ou enlever des points d'arrêt, vous pouvez également utiliser la commande 'En Ligne' 'Point d'arrêt actif/inactif'.

Boîte de dialogue pour éditer les points d'arrêt

'En Ligne' 'Étape individuelle sur'

Icône : **Raccourci : <F10>**

Cette commande permet d'exécuter une étape individuelle. Dans le cas d'un appel de module, l'exécution s'arrête uniquement après que l'exécution du module est terminée. Dans l'éditeur SFC, une action complète est exécutée.

Lorsque l'instruction actuelle consiste en un appel de fonction ou de bloc fonctionnel, la fonction ou le bloc fonctionnel sont exécutés jusqu'au bout. Pour aller à la première instruction d'une fonction ou d'un bloc fonctionnel qui a été appelé(e), utilisez la commande 'En ligne' 'Etape individuelle dans'.

Lorsque la dernière instruction est atteinte, le programme passe à l'instruction suivante du module à partir duquel l'appel a été effectué.

'En Ligne' 'Étape individuelle dans'

Raccourci : <F8>

Cette commande permet d'exécuter une étape individuelle. Dans le cas de l'appel de modules, l'application s'arrête avant l'exécution de la première instruction du module.

Le cas échéant, vous basculez dans un module qui est appelé.

Lorsque la position actuelle consiste en un appel de fonction ou de bloc fonctionnel, le programme passe à de la première instruction du module appelé.

Dans les autres cas, cette commande se comporte exactement comme la commande 'En ligne' 'Étape individuelle sur'.

'En Ligne' 'Cycle indépendant'

Raccourci : <Ctrl>+<F5>

Cette commande se limite à effectuer un cycle de commande unique, après quoi il s'arrête.

Cette commande peut être répétée continuellement, pour avancer en effectuant des cycles uniques.

Un cycle indépendant prend fin lorsque que la commande 'En ligne' 'Démarrer' est exécutée.

'En Ligne' 'Écrire valeurs des variables'

Raccourci : <Ctrl>+<F7>

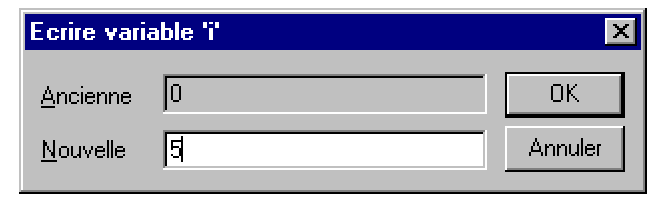
Cette commande permet de ramener au début d'un cycle - une seule fois ! - une ou plusieurs variable(s) à des valeurs définies par l'utilisateur.

Vous pouvez modifier la valeur de toutes les variables composées d'un seul élément qui sont aussi visibles à l'espionnage.

Avant d'effectuer la commande 'Ecrire valeurs des variables', vous devez définir une valeur de variable pour l'écriture.

<u>1. Définition des valeurs :</u>

- Pour les variables autres que booléennes, il suffit de double-cliquer sur la ligne à laquelle la variable est déclarée, ou de marquer cette variable et d'appuyer sur la touche <Entrée>. Suite à quoi une boîte de dialogue 'Écrire variable <x>' apparaît, vous permettant de saisir la valeur qui sera écrite dans la variable.
  Boîte de dialogue pour l'écriture d'une nouvelle valeur de variable



- Pour les variables booléennes, il suffit de double-cliquer sur la ligne à laquelle la variable est déclarée, et la valeur change d'un état à l'autre (TRUE, FALSE, et pas de nouvelle valeur) sans qu'une boîte de dialogue apparaisse.

La nouvelle valeur prévue pour l'écriture est affichée couleur turquoise entre les signes '<' et '>', à la suite de l'ancienne valeur, par exemple a=0 <:=34>.

bvar = TRUE < := FALSE>
ivar = 509 < := 65>

Remarque : Exception à l'affichage des valeurs à écrire : Dans les éditeurs FBD et LD, la nouvelle valeur est indiquée en couleur turquoise mais sans les signes '<' et '>'.

Vous pouvez définir une nouvelle valeur pour autant de variables que vous le souhaitez.

Les valeurs introduites pour les variables à écrire peuvent également être corrigées voire effacées de la même manière. Ceci est également possible avec 'En Ligne' 'Dialogue Écrire/Forcer' (voir ci-dessous).

Les valeurs attribuées pour l'écriture sont sauvegardées dans une **liste d'écriture** (**Liste d'espion**) jusqu'à leur utilisation effective ou leur effacement; elles peuvent également être déposées dans une liste de forçage via la commande 'Forcer valeurs des variables' Les listes d'espion et de forçage peuvent être consultées dans 'Dialogue Ecrire/Forcer'.

<u>2. Écriture des valeurs :</u>

La commande permettant l'écriture des valeurs provenant de la liste d'écriture se trouve en deux endroits :

- Commande 'Écrire valeurs des variables' dans le menu 'En Ligne'.
- Bouton <Écrire valeurs des variables> dans la boîte de dialogue 'Editer la liste d'écriture et de forçage'.

Si vous exécutez la commande 'Écrire valeurs des variables', toutes les valeurs contenues dans la liste d'écriture sont attribuées une seule fois en début de cycle aux variables correspondantes de l'automate et ainsi effacées de la liste d'écriture. (Si la commande 'Forcer valeurs des variables' est exécutée, les valeurs sont également effacées de la liste d'écriture et reprises dans la Liste de forçage!)

Remarque : Dans le diagramme fonctionnel en séquence, les valeurs individuelles formant une expression de transition ne peuvent être modifiées par le biais de 'Écrire des valeurs'. Ceci repose sur le fait que lors de l'espionnage, la valeur totale de l'expression est représentée, et non pas la valeur des variables individuelles (p.ex. « a ET b » ne seront représentés comme TRUE que si les deux valeurs ont effectivement la valeur TRUE).

En FBD par contre, la première variable est seule espionnée dans une expression utilisée par exemple comme entrée d'un bloc fonctionnel. Ainsi, 'Écrire valeurs des variables' est possible pour cette seule variable.

'En Ligne' 'Forcer valeurs des variables'

Raccourci : <F7>

Cette commande permet de ramener de manière permanente une ou plusieurs variable(s) à des valeurs définies par l'utilisateur. Cette opération s'effectue dans le système d'exécution au début et à la fin d'un cycle.

<u>Déroulement chronologique au sein d'un cycle</u> : 1.lire les entrées, 2. forcer les valeurs, 3. exécuter le code, 4. forcer les valeurs, 5. écrire les sorties.

La fonction est active tant qu'elle n'est pas expressément interrompue par l'utilisateur (Commande 'En Ligne' 'Arrêter de forcer') ou que le système de programmation ait fermé la session.

Pour la création d'une nouvelle valeur, une **Liste d'espion** (liste d'écriture) est tout d'abord créée, comme cela a été décrit sous 'En Ligne' 'Ecrire valeurs des variables' (voir 1. Définition des valeurs). Les variables contenues dans la liste d'écriture sont identifiées en conséquence dans l'espion. La liste d'écriture est transmise dans une **Liste de forçage** dès que la commande 'Forcer valeurs des variables' est exécutée. Les entrées dans la liste de forçage peuvent être consultées dans 'Dialogue Écrire/Forcer'. Il se peut qu'il y ait déjà une Liste de forçage active, celle-ci sera alors mise à jour en conséquence. La liste d'écriture se vide et les nouvelles valeurs sont représentées en rouge comme "forcées"; par exemple:

```
bvar = TRUE <:= FALSE>
ivar = 509 <:= 65>
```

Des modifications dans la Liste de forçage sont transmises au programme lors d'une commande ultérieure 'Forcer valeurs des variables'.

Veuillez noter : La Liste de forçage est créée lors du premier forçage des valeurs contenues dans la Liste d'espion alors que cette dernière existait déjà <u>avant</u> la première écriture des valeurs qu'elle contient.

Remarque: Si le système cible le supporte, une liste de forçage est maintenue sur l'automate programmable, même si la connexion est interrompue p.ex. par une fin de session.

La commande permettant de forcer une variable (et par là même l'enregistrement dans une liste de forçage) se trouve aux emplacements suivants :

- Commande 'Forcer valeurs des variables' dans le menu 'En Ligne'.

- Bouton <Forcer valeurs des variables> dans la boîte de dialogue 'Editer la liste d'écriture et de forçage'.

Remarque : Dans le diagramme fonctionnel en séquence, les valeurs individuelles permettant de former une expression de transition ne peuvent être modifiées par le biais de 'Forcer valeurs des variables'. Ceci repose sur le fait que lors de l'espionnage, la valeur totale de l'expression est représentée, et non pas la valeur des variables individuelles (p.ex. « a ET b » ne seront représentés comme TRUE que si les deux valeurs ont effectivement la valeur TRUE).
En FBD par contre, la première variable est seule espionnée dans une expression utilisée par exemple comme entrée d'un bloc fonctionnel. Ainsi, 'Forcer valeurs des variables' n'est possible que pour cette variable.

Dans le diagramme fonctionnel en séquence, les valeurs individuelles formant une expression de transition ne peuvent être modifiées par le biais de 'Forcer valeurs des variables'. Ceci repose sur le fait que lors de l'espionnage, la valeur totale de l'expression est représentée, et non pas la valeur des variables individuelles (p.ex. « a ET b » ne seront représentés comme TRUE que si les deux valeurs ont effectivement la valeur TRUE).

En FBD par contre, la première variable est seule espionnée dans une expression utilisée par exemple comme entrée d'un bloc fonctionnel. Ansi, 'Forcer valeurs des variables' n'est possible que pour cette variable.

'En Ligne' 'Arrêter de forcer'

Raccourci : <Maj>+<F7>

Cette commande met fin au Forçage de valeurs de variables dans l’automate. Les variables modifient à nouveau leur valeur selon la procédure normale.

Les variables forcées sont reconnaissables à leur représentation en rouge dans l’espion. Vous avez la possibilité d’effacer complètement la liste de forçage ou d’en effacer quelques variables individuelles (à sélectionner avant de donner la commande d'arrêt de forçage).

Pour effacer complètement la liste de forçage et donc lever le forçage pour **toutes les variables**, sélectionnez les possibilités suivantes :

- Commande 'Arrêter de forcer' dans le menu 'En Ligne'.
- Bouton <Arrêter de forcer> dans la boîte de dialogue 'Editer la liste d'écriture et de forçage'.
- Effacer toute la liste de forçage par le biais de la boîte de dialogue 'Effacer les listes d'écriture/de forçage'. Celle-ci s'ouvre grâce à la commande 'En Ligne' 'Arrêter de forcer' pour autant qu’il y ait encore des entrées dans la Liste d'espion.

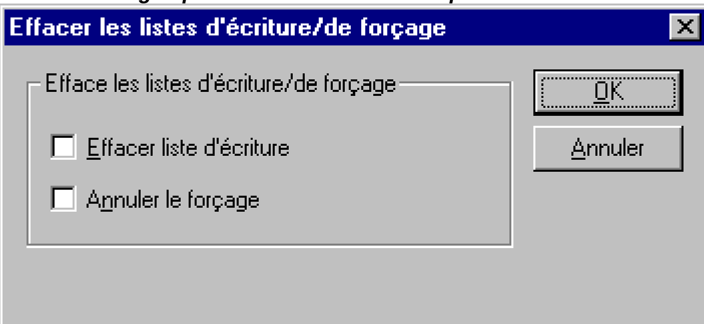
Pour lever le forçage de **quelques variables individuelles** de la 'Liste de forçage', vous devez tout d’abord marquer ces variables. Vous devez pour ce faire sélectionner une des possibilités décrites ci-après. Les variables concernées sont reconnaissables à l’ajout couleur turquoise <Arrêter de forcer>.

- Un double-clic sur la ligne à laquelle une variable non booléenne à forcer est déclarée donne accès à la boîte de dialogue 'Écrire variable <x>'. Appuyez alors sur le bouton <Arrêter de forcer pour cette variable>.
- En multipliant les doubles-clics sur la ligne à laquelle une variable booléenne à forcer est déclarée, vous pouvez modifier les données jusqu’à l’indication <Arrêter de forcer> après la variable.
- Effacez la valeur dans le champ d’édition de la colonne 'Valeur forcée' dans la boîte de dialogue Écrire/Forcer, à laquelle vous avez accès via le menu 'En Ligne'.

Si '<Arrêter de forcer>' se trouve après la valeur de toutes les variables souhaitées dans la fenêtre de déclaration, vous pouvez exécuter la commande 'Forcer valeurs des variables', qui transmettra le nouveau contenu de la Liste de forçage à l’automate.

Si, lors de l’exécution de la commande 'Arrêter de forcer', la Liste d'espion (voir 'En Ligne' 'Écrire valeurs des variables') n’est pas vide, la boîte de dialogue 'Effacer les listes d'écriture/de forçage' s’ouvre, incitant l’utilisateur à décider s’il veut **Annuler le forçage** ou **Effacer la liste d'écriture**, ou les deux.

Boîte de dialogue pour effacer les listes d'espion et d'écriture



'En Ligne' 'Dialogue Ecrire/Forcer'

Cette commande donne accès à une boîte de dialogue représentant sous deux onglets la liste d'écriture actuelle (**Liste d'espion**) ainsi que la **Liste de forçage**. Les noms de variables ainsi leurs valeurs préparées ou forcées pour l'écriture sont toutes représentées dans un tableau.

Dialogue pour Éditer la liste d'écriture et de forçage

Editer la liste d'écriture et de forçage

Liste d'espion | Liste de forçage | OK | Annuler

Variable	Valeur préparée
.GEO.iSourceLine_No	44
.GEO.iObjNo	44

Editer la liste d'écriture et de forçage

Liste d'espion | Liste de forçage | OK | Annuler | Ecrire valeurs des variables | Forcer valeurs des variables | Arrêter de forcer

Variable	Valeur forcée
.GEO.iSourceLine_No	3

Grâce à la commande 'En Ligne' 'Écrire valeurs des variables', les valeurs parviennent à la 'Liste d'espion', et elles sont transmises dans la 'Liste de forçage' par le biais de la commande 'En Ligne' 'Forcer valeurs des variables'. Les valeurs peuvent être éditées dans les colonnes 'Valeur préparée' ou 'Valeur forcée' en ce sens qu'un clic de la souris sur l'entrée ouvre un champ d'édition. Un message d'erreur est délivré en cas d'inconsistance de type. Si vous effacez une valeur, cela signifie que l'entrée est enlevée de la liste d'écriture ou que la variable est marquée pour lever le forçage dès que vous quittez la boîte de dialogue par le biais d'une commande autre que **Annuler**.

Les commandes suivantes, correspondant à celles du menu En ligne, sont à disposition par le biais des boutons :

Forcer valeurs des variables: Toutes les entrées de la liste d'écriture actuelle sont déplacées dans la liste de forçage, ce qui signifie que les valeurs de variables sont forcées dans l'automate. Les variables marquées par "<Arrêter de forcer>" ne sont plus forcées. La boîte de dialogue se referme ensuite.

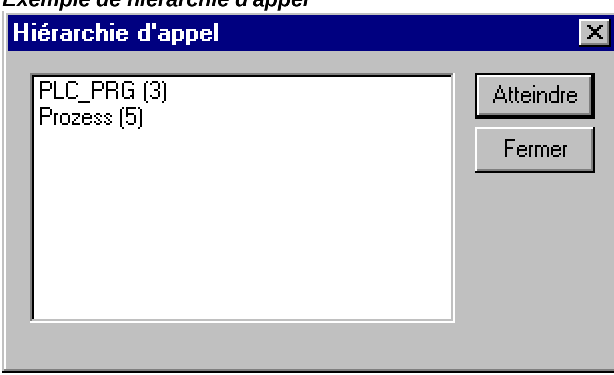
Ecrire valeurs des variables: Toutes les entrées de la liste d'écriture actuelle sont écrites une seule fois dans les variables correspondantes dans l'automate. La boîte de dialogue se referme ensuite.

Arrêter de forcer: Toutes les entrées de la liste de forçage sont effacées ou, si une liste d'écriture est disponible, la boîte de dialogue 'Effacer les listes d'écriture/de forçage' s'ouvre, incitant l'utilisateur à décider s'il veut **Annuler le forçage** ou **Effacer la liste d'écriture**, ou les deux. La boîte de dialogue se referme ensuite ou après avoir fermé la boîte de dialogue qui a permis la sélection.

'En Ligne' 'Hiérarchie d'appel'

Vous pouvez appeler cette commande lorsque la simulation s'arrête à un point d'arrêt. Une boîte de dialogue s'ouvre et affiche une liste des modules qui se trouvent actuellement dans la pile d'appel.

Exemple de hiérarchie d'appel



Le premier module est toujours PLC_PRG, car c'est dans ce module que débute l'exécution.

Le dernier module est toujours celui où le programme était exécuté actuellement.

Après avoir sélectionné un des modules et cliqué sur le bouton **Atteindre**, le module sélectionné est chargé dans une fenêtre, et la ligne ou le réseau en cours d'exécution est affiché(e).

'En Ligne' 'Contrôle de déroulement' '

Selon la configuration du système cible actuel, on peut activer ou désactiver le contrôle de déroulement par l'utilisateur au moyen de cette option de menu. Si **Contrôle de déroulement** est sélectionné, un crochet (✓) apparaît devant l'élément de menu. Tous les réseaux ou lignes qui ont été exécutés au cours du dernier cycle de commande sont marqués.

Les champs de numérotation des lignes ou des réseaux qui ont été parcourus sont visualisés en vert. Dans l'éditeur IL, un champ supplémentaire est ajouté au niveau du bord gauche de chaque ligne. Le contenu actuel de l'accumulateur y est affiché. Dans les éditeurs graphiques du schéma en blocs fonctionnels et du langage à contacts, un champ supplémentaire est ajouté à tous les éléments de liaison qui ne véhiculent pas de valeurs booléennes. Lorsque ces entrées et sorties sont définies, la valeur véhiculée par l'élément de liaison s'affiche dans ce champ. Les éléments de liaison qui transportent exclusivement des valeurs booléennes apparaissent en bleu lorsqu'ils transportent des valeurs TRUE; ce système permet de suivre continuellement le flux d'informations.

'En Ligne' 'Simulation'

Si la **Simulation** est sélectionnée, un crochet (✓) apparaît devant l'élément de menu.

En mode simulation, le programme utilisateur est exécuté à même le PC sous Windows. Le mode simulation sert à tester le projet. La communication entre le PC et la simulation utilise le technique usuelle de transfert de messages de Windows.

Lorsque le programme n'est pas en mode simulation, il est exécuté sur l'automate programmable. La communication entre le PC et l'automate programmable se fait de manière usuelle, par l'intermédiaire de l'interface série.

L'état de ce drapeau est enregistré avec le projet.

'En Ligne' 'Paramètres de communication'

Vous accédez au moyen de cette commande à une boîte de dialogue destinée à configurer les paramètres de communication entre l'ordinateur local et le système d'exécution via une gateway. (Si vous utilisez un serveur OPC ou DDE, les mêmes paramètres de communication doivent y être configurés.)

Voyez à cet effet les points suivants :

- Principe de la gateway
- Boîte de dialogue "Paramètres de communication" de l'ordinateur local
- Configuration de la passerelle souhaitée et du canal
- Création d'un nouveau canal pour la passerelle locale
- Indications pour éditer les paramètres dans la boîte de dialogue Paramètres de communication

Principe de la gateway

Votre ordinateur local peut être relié à un ou plusieurs systèmes d'exécution via une gateway. Une configuration particulière pour chaque gateway détermine les systèmes d'exécution auxquels la gateway peut accéder tandis que la liaison avec la gateway voulue est configurée au niveau de l'ordinateur local. Il est possible de créer une configuration telle que, non seulement la gateway, mais aussi un ou plusieurs systèmes d'exécution soient en exécution sur l'ordinateur local. Si la gateway est installée sur l'ordinateur local, l'échange entre le système de programmation et la gateway se fait par le biais de la mémoire partagée ou de TCP/IP. S'il s'agit d'une gateway qui est en exécution sur un PC étranger, il faut s'assurer qu'elle a démarré sur ce même PC. La connexion n'est alors possible que par le biais de TCP/IP.

La gateway démarre automatiquement dès que, sur l'ordinateur dans lequel il est installé, la boîte de dialogue permettant la configuration de la communication s'ouvre dans le cadre de **CoDeSys**, ou dès que vous vous connectez au système d'exécution cible. Si une version de la gateway incompatible avec le système de programmation est installée sur votre ordinateur, le message correspondant est affiché. Il est alors impossible d'ouvrir une session.

Une icône CoDeSys apparaît alors en bas à droite, dans la barre des tâches. Cette icône est allumée aussi longtemps que vous êtes relié au système d'exécution via la gateway.

Le menu de passerelle :

En cliquant sur le bouton droit de la souris sur ladite icône, vous accédez aux options **Aide, Info**, **Changer mot de passe**,**Inspection**, et **Quitter**.

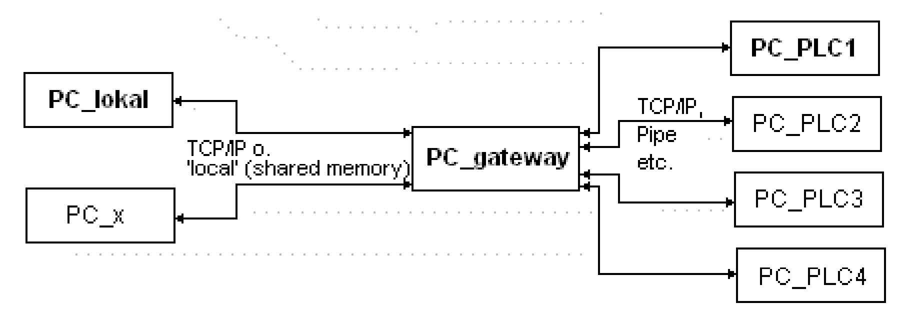
Vous obtenez des informations à propos de la version de la gateway par le biais d'**Info**.

Par le biais de **Changer mot de passe**, vous obtenez une boîte de dialogue vous permettant de saisir un mot de passe pour la gateway local ou éventuellement de changer ce mot de passe. Si une telle protection existe, vous êtes invité à saisir le mot de passe dès que la gateway concernée est sélectionnée dans la boîte de dialogue des paramètres de communication ou dès que vous vous connectez pour la première fois.

Par le biais de **Inspection**, vous accédez aux boîtes de dialogue du contrôleur de gateway, permettant un espionnage des canaux de gateway (canaux disponibles, services actifs, etc.). Vous pouvez ouvrir, via l'option **Aide**, l'aide En ligne de l'interface utilisateur de la gateway, afin d'obtenir des informations quant à l'utilisation du contrôleur.

La commande **Quitter** vous permet de déconnecter la gateway.

Le schéma suivant représente un système avec gateway:



PC_lokaldésigne votre ordinateur local tandis que **PC_x** désigne un PC distinct de l'ordinateur local, utilisant la gateway. **PC_gateway** désigne le PC sur lequel est installée la gateway, **PC_PLC1**, **PC_PLC2**, **PC_PLC3** et **PC_PLC4** désignent les PC sur lesquels les systèmes d'exécution sont en exécution. Les modules apparaissent séparés sur la figure, mais il est tout à fait possible que la gateway et / ou le système d'exécution soient également installés sur l'ordinateur local.

Attention : Nous attirons votre attention sur le fait que la liaison avec la passerelle ne peut se faire qu'en utilisant le protocole TCP/IP. Par conséquent, votre ordinateur doit être équipé de façon adéquate. Si par contre la passerelle est installée sur votre ordinateur local, la liaison est possible via la mémoire partagée (série).Les liaisons entre la passerelle et les différents ordinateurs d'exécution peuvent quant à elles utiliser divers protocoles (TCP/IP, Pipe, etc.).

Représentation dans la boîte de dialogue 'Paramètres de communication'

Cette boîte de dialogue sert à sélectionner une gateway par l'intermédiaire de laquelle la connexion doit par exemple s'effectuer avec un automate. En outre, vous pouvez établir de nouveaux canaux pour une gateway installée sur l'ordinateur local et définir leurs paramètres de connexion, de manière à ce que ceux-ci soient également disponibles pour d'autres ordinateurs du réseau.

Vous pouvez à tout moment appeler les paramètres actuellement en vigueur via le bouton **Actualiser**.

Si les paramètres de communication ont déjà été configurés conformément au schéma présenté à titre d'exemple 'Principe de la gateway', la boîte de dialogue se présenterait de la façon suivante:

Boîte de dialogue Paramètres de communication par gateway, exemple de configuration

Paramètres de communication

Canaux
- localhost
  - PC_PLC1
  - PC_PLC3
  - PC_PLC2
- PC_PLC1
- PC_PLC4

Tcp/Ip | 3S Tcp/Ip driver

Nom	Valeur	Commentaire
Address	localhost	IP address or hostname
Port	1200	
Motorola byteor...	No	

OK | Annuler | Nouveau ... | Effacer | Gateway ... | Actualiser

La rubrique **Canaux** affiche deux catégories de liaisons:

- La première catégorie regroupe tous les canaux que la gateway actuellement connectée, désignée par "localhost", offre sur le réseau, permettant par exemple une liaison avec l'automate. L'adresse ou le nom de ladite gateway est indiqué après le signe moins, à l'extrémité supérieure. Dans l'exemple que nous avons choisi, cette gateway est en exécution sur l'ordinateur local. L'adresse appropriée "localhost" correspond en règle générale à l'adresse IP 127.0.0.1 de l'ordinateur local (PC_lokal). En dessous, trois adresses de systèmes d'exécution sont indentées au sein d'une arborescence. Des canaux sont configurés pour ces systèmes d'exécution, au niveau de la gateway (PC_PLC1 à 3). Ces canaux ont pu être configurés aussi bien par l'ordinateur local que par d'autres PC (PC_x), qui sont ou ont été reliés à la gateway.
- La deuxième catégorie de canaux englobe toutes les liaisons au niveau de la gateway ayant été configurées à partir de votre ordinateur local – par exemple, à l'aide de cette boîte de dialogue de configuration. Ces liaisons constituent la "branche" qui descend en droite ligne du signe moins vers PC_PLC1 et PC_PLC4. Les adresses de ces canaux ne doivent pas nécessairement avoir déjà été identifiées au niveau de la gateway. Certes, les paramètres de communication pour PC_PLC4, dans l'exemple représenté ci-dessus, sont enregistrés localement dans le projet. Néanmoins, ces paramètres n'ont été identifiés par la gateway qu'après un nouvel accès au système d'exécution. C'est le cas pour PC_PLC1, qui a été rajouté pour cette raison à "l'arborescence des canaux", comme "sous-branche" de l'adresse de gateway concernée.

Dans la partie centrale de la boîte de dialogue se trouvent l'identification du canal sélectionné dans la partie de gauche ainsi que les paramètres qui se rapportent à ce canal, dans les rubriques **Nom**, **Valeur** et **Commentaire**.

Configuration de la gateway souhaitée et du canal

1. Sélection de la passerelle dans le dialogue Paramètres de communication:

Pour définir la liaison avec la gateway voulue, ouvrez la boîte de dialogue Paramètres de communication par Gateway en appuyant sur le bouton **Gateway**.

Exemple de boîte de dialogue, Définition de la connexion locale à la passerelle

Paramètres de communication: Gateway
Connection: Tcp/Ip
Adresse: localhost
Mot de passe:
Port: 1210
OK
Annuler

Vous pouvez alors saisir ou éditer les données suivantes :

- le type de **connexion** que vous souhaitez utiliser entre votre ordinateur et l'ordinateur sur lequel la gateway est en exécution. Si la gateway est en exécution sur l'ordinateur local, une connexion via la mémoire partagée ("Local") est possible, de même qu'une connexion par le biais de TCP/IP; si vous devez vous connecter à un autre ordinateur, vous ne pouvez utiliser que TCP/IP.
- l'**adresse** de l'ordinateur que vous souhaitez utiliser et sur lequel la gateway tourne. Adresse IP ou nom symbolique correspondant, par exemple localhost. Lors d'une première configuration, « localhost » est proposé par défaut comme nom d'ordinateur (adresse), ce qui signifie qu'on accéderait à la gateway installée localement. Le nom "localhost" est automatiquement assimilé dans la plupart des cas à l'adresse IP locale 127.0.0.1. Cependant, il se peut que vous ayez à entrer directement cette dernière dans le champ Adresse. Si vous souhaitez accéder à une gateway sur un autre ordinateur, vous devez remplacer « localhost » par son nom ou son adresse IP.
- le **mot de passe** pour la gateway sélectionnée, si celle-ci est installée sur un ordinateur à distance. Si celui-ci est erroné ou que vous ne l'introduisez pas, un message d'erreur apparaît à

l'écran.
Veuillez noter à cet effet : vous pouvez munir d'un mot de passe la gateway installée localement de la manière suivante : Cliquez avec le bouton droit de la souris sur l'icône de la gateway en bas à droite dans la barre des tâches, et sélectionnez « Changer mot de passe ». Vous accédez alors à une boîte de dialogue permettant de changer ou de saisir un mot de passe. Si vous accédez localement à la gateway, le mot de passe éventuellement attribué n'est pas demandé.

- le **port** de l'ordinateur que vous souhaitez utiliser et sur lequel la gateway tourne ; en règle générale, il est déjà prédéfini pour la valeur correspondant à la gateway sélectionnée.

Si vous fermez la boîte de dialogue en confirmant par **OK**, l'entrée correspondante (adresse d'ordinateur) apparaît tout en haut dans la rubrique **Canaux** de la boîte de dialogue Paramètres de communication, et les canaux disponibles de cette gateway apparaissent en dessous.

2. Configuration du canal souhaité de la gateway sélectionnée

Sélectionnez maintenant un des canaux en cliquant sur une des entrées. Les paramètres correspondants sont alors affichés dans un tableau. Si aucune connexion ne peut être créée vers l'adresse de gateway choisie - du fait que le serveur ne soit pas démarré ou que l'adresse soit erronée , alors l'indication "non connecté" apparaît entre parenthèses derrière l'adresse ainsi que le message 'Aucune gateway avec ces paramètres n'a été trouvée'. Reportez-vous dans ce cas à la rubrique 'Vérification en cas d'échec de connexion à la gateway'.

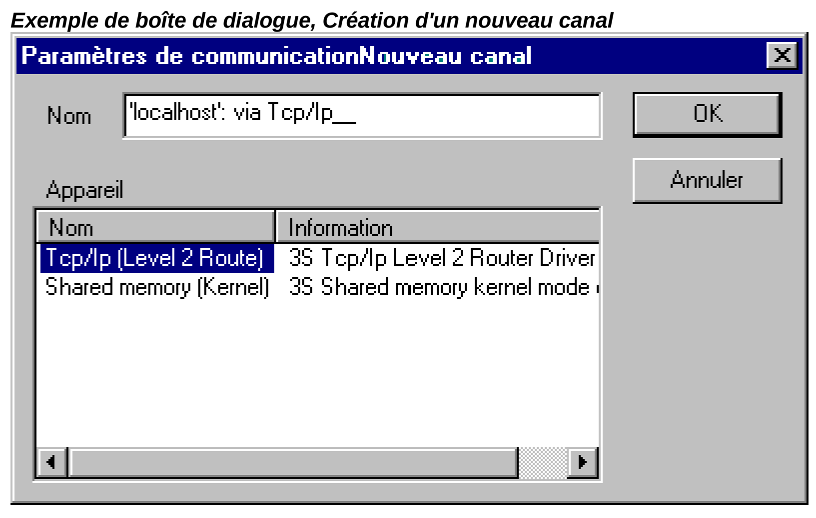
Lorsque le canal souhaité est configuré, fermez cette boîte de dialogue avec **OK**. Les configurations sont enregistrées avec le projet.

Création d'un nouveau canal pour la gateway locale

Dans le dialogue Paramètres de communication, vous pouvez créer des nouveaux canaux pour la passerelle actuellement connectée, ceux-ci seront alors à disposition pour les connexions de la passerelle, par exemple pour une connexion avec l'automate. Les possibilités dont vous disposez sont fonction de la gamme individuelle des pilotes installés sur votre ordinateur.

Appuyez sur le bouton **Nouveau...**. Vous obtenez alors la boîte de dialogue 'Paramètres de communication :Nouveau canal' :

Exemple de boîte de dialogue, Création d'un nouveau canal



- Le nom correspondant au dernier canal entré est automatiquement proposé dans le champ **Nom**. Si aucun canal n'a été défini jusque là, le nom de la gateway actuelle est proposé suivi d'un caractère de soulignement, par exemple "localhost_". A cet endroit, vous pouvez modifier le nom de canal. Les noms de canaux sont purement indicatifs. Il n'est donc pas nécessaire qu'ils soient univoques, même si cela est préférable.
- Les pilotes disponibles auprès de l'ordinateur gateway sont énumérés dans un tableau sous **Appareil**. En cliquant avec la souris, sélectionnez un des pilotes proposés dans la colonne **Nom** ; s'il existe, le commentaire correspondant est affiché dans la colonne **Information**.

Si vous fermez la boîte de dialogue 'Nouveau canal' en confirmant par **OK**, le canal nouvellement défini est affiché dans la boîte de dialogue Paramètres de communication, dans la rubrique **Canaux**, en dernière position sous le signe moins. Pour le moment, le canal n'est enregistré que localement dans le projet (voir ci-dessus) ! À ce stade, vous pouvez éditer la colonne **Valeur**. Confirmez les paramètres réglés à l'aide d'**OK** et quittez la boîte de dialogue 'Paramètres de communication'.

Vous devez **accéder au système d'exécution** afin que le nouveau canal de gateway et ses paramètres soient identifiés au niveau de la gateway xy et afin qu'il soit accessible à tous les ordinateurs qui ont accès à cette gateway xy. Si vous ouvrez ensuite à nouveau la boîte de dialogue 'En Ligne' 'Paramètres de communication' le nouveau canal apparaît dans une "arborescence de canaux" et en indentation sous l'adresse / le nom de l'ordinateur sur lequel la gateway est installée, tout en continuant à être affiché à la position qu'il occupait jusque-là. Ceci signifie qu'il est connu du réseau. Vous pouvez alors ouvrir la boîte de dialogue des paramètres de communication à partir d'un autre ordinateur que l'ordinateur local, sélectionner la gateway xy et utiliser son nouveau canal.

Si une erreur de communication est affichée lors de l'accès au système, alors il se peut qu'il soit impossible d'ouvrir l'interface (par exemple COM1 dans le cas d'une liaison série). En effet, il se peut que cette interface soit déjà occupée par un autre périphérique. Il se peut également que seul l'automate ne fonctionne pas.

Il ne vous est plus possible d'éditer, à partir de la boîte de dialogue de configuration, les paramètres d'un canal déjà identifié par la gateway. Les champs de paramètres sont visualisés en gris. En revanche, vous conservez la possibilité de supprimer la liaison, à condition que celle-ci ne soit pas active.

Attention : Nous attirons votre attention sur le fait que, dans le cas d'un canal, l'opération de suppression ne soit pas réversible. La suppression est effectuée au moment où vous appuyez sur le bouton **Effacer** !

Indications pour éditer les paramètres dans la boîte de dialogue Paramètres de communication

Dans le dialogue Paramètres de communication, vous pouvez éditer uniquement les champs de texte de la colonne **Valeur**.

En sélectionnant un champ de texte avec la souris, vous pouvez accéder au mode édition à l'aide d'un double-clic ou en appuyant sur la <Barre d'espacement>. Pour terminer une entrée de texte donnée, appuyez sur la touche <Entrée>.

Pour vous déplacer vers le bouton ou la zone d'édition qui suit ou qui précède, appuyez respectivement sur la touche <Tabulation> et sur les touches <Maj> + <Tabulation>.

Si vous éditez des valeurs numériques, il vous est possible d'incrémenter ou de décrémenter une valeur donnée d'une ou de dix unités, à l'aide des touches directionnelles ou des touches de défilement d'écran. En double-cliquant avec la souris, vous incrémentez également la valeur d'une unité. Pour les valeurs numériques, une vérification relative au type a été configurée: en appuyant sur les touches <Ctrl> + <Pos1> et <Ctrl> + <Fin>, vous obtenez respectivement la valeur minimale et la valeur maximale susceptible d'être entrée, pour le type de paramètre choisi.

Vérification en cas d'échec de connexion à la gateway

Vérifiez les points suivants si vous ne parvenez pas à établir une connexion avec la gateway sélectionnée (Le message 'Non connecté' apparaît dans la boîte de dialogue des paramètres de communication derrière l'adresse de la gateway, dans le champ Canaux) :

- La gateway est-elle démarrée (apparition d'un symbole tricolore dans le coin inférieur droit dans la barre des tâches) ?
- S'agit-il réellement, au niveau de l'adresse IP que vous avez saisi dans la boîte de dialogue 'Gateway: 'Paramètres de communication, de l'ordinateur sur lequel la gateway est en exécution ? (Vérifier au moyen d'un « ping »)

La connexion TCP/IP fonctionne localement ? L'erreur se situe éventuellement au niveau du TCP/IP.

'En Ligne' 'Application téléchargée du code source dans l'automate'

Cette commande permet de charger le code source du projet dans l'automate programmable. À ne pas confondre avec le code créé à la compilation du projet ! Vous pouvez déterminer les options

valables pour un téléchargement (moment, taille) dans la boîte de dialogue 'Projet' 'Options' 'Téléchargement code source'.

'En Ligne' 'Créer projet d'initialisation'

Si cette commande est effectuée en ligne, le projet compilé est complètement sauvegardée dans l'automate de sorte que ce dernier puisse le charger lors d'un nouveau démarrage. La sauvegarde du projet d'initialisation se produit de différentes façons, en fonction du système cible. Par exemple, trois fichiers sont créés sur les systèmes basés sur un processeur 386 : default.prg contient le code projet, default.chk le total de contrôle du code, et default.sts le statut de l'automate après un nouveau démarrage (Start/Stop).

La commande 'En Ligne' 'Créer projet d'initialisation' est également à disposition **Hors ligne**, lorsque le projet a été compilé sans erreurs. Dans ce cas, le fichier **<Nom du projet>.prg** est créé pour le projet d'initialisation dans le répertoire du projet, et le fichier **<Nom du projet>.chk** pour le total de contrôle du code. Il peuvent être copiés sur un automate après en avoir changé le nom de façon adéquate.

S'il existe déjà un projet d'initialisation dans l'automate programmable et si en outre l'option 'Mode En ligne en mode de sécurité' est activée au sein des options de projet, catégorie Environnement de travail, un dialogue s'ouvre lors de la création d'un nouveau projet d'initialisation, reprenant les **informations relatives au projet** actuellement chargé sur le système de programmation et au projet d'initialisation existant au niveau de l'automate. Cette fonctionnalité doit bien entendu être supportée par le système cible !

Toujours selon la configuration du système cible, un nouveau fichier *.ri (informations de compilation) sera éventuellement créé en même temps que la création Hors ligne d'un projet d'initialisation. Si un tel fichier existe, un dialogue de confirmation s'ouvrira le cas échéant (selon le système cible).

Remarque : Si l'option Automatiquement à la création du projet d'initialisation (catégorie Download Sourcecode) est activée, la commande 'En Ligne' 'Créer projet d'initialisation' vous permet de charger automatiquement dans l'automate programmable les fichiers source choisis.

'En Ligne' 'Ecrire le fichier dans l'automate'

Cette option sert à charger un fichier au choix dans l'automate. Elle donne accès à la boîte de dialogue 'Écrire un fichier dans l'automate' dans laquelle vous marquez le fichier souhaité. Après avoir confirmé la sélection au moyen du bouton **Ouvrir**, le dialogue se referme, le fichier est chargé dans l'automate puis y est créé sous le même nom. Le chargement s'accompagne d'une barre d'avancement.

La commande 'En Ligne' 'Charger le fichier de l'automate' permet de charger à nouveau un fichier déjà créé dans l'automate.

'En Ligne' 'Charger le fichier de l'automate'

Cette commande permet de charger à nouveau un fichier créé dans l'automate par le biais de 'En Ligne' 'Ecrire le fichier de l'automate'. Vous obtenez la boîte de dialogue 'Charger le fichier de l'automate'. Introduisez le nom du fichier souhaité sous **Nom du fichier** et sélectionnez le répertoire dans lequel il doit être chargé (**Enregistrer sous**) dès que vous avez refermé la boîte de dialogue via le bouton 'Enregistrer'.

4.7 Ordonner les fenêtres

Dans l'option '**Fenêtre**' se trouvent toutes les commandes relatives à la gestion des fenêtres. Il s'agit aussi bien de commandes pour ordonner automatiquement vos fenêtres que pour ouvrir le gestionnaire de bibliothèques et pour basculer d'une fenêtre ouverte vers une autre fenêtre ouverte.

A la fin du menu toutes les fenêtres ouvertes sont répertoriées dans l'ordre dans lequel elles ont été ouvertes. En cliquant avec la souris sur l'entrée appropriée, vous accédez à la fenêtre de votre choix. Un crochet (✓) apparaît devant le nom de la fenêtre active.

Les paragraphes qui suivent décrivent en détail les commandes du menu 'Fenêtre':

'Fenêtre' 'Mosaïque horizontale'

Cette commande vous permet d'ordonner l'une à côté de l'autre toutes les fenêtres à l'intérieur de l'environnement de travail, de façon à ce qu'elles ne se recoupent pas et qu'elles occupent la totalité de l'environnement de travail.

'Fenêtre' 'Mosaïque verticale'

Cette commande vous permet d'ordonner l'une au-dessus de l'autre toutes les fenêtres à l'intérieur de l'environnement de travail, de façon à ce qu'elles ne se recoupent pas et qu'elles occupent la totalité de l'environnement de travail.

'Fenêtre' 'Cascade'

Cette commande vous permet d'ordonner l'une derrière l'autre toutes les fenêtres à l'intérieur de l'environnement de travail, de façon à former une cascade.

'Fenêtre' 'Réorganiser les icônes'

Cette commande vous permet d'ordonner toutes les fenêtres réduites, situées à l'intérieur de l'environnement de travail, sur une ligne située à la limite inférieure de l'environnement de travail.

'Fenêtre' 'Fermer tout'

Cette commande vous permet de fermer toutes les fenêtres ouvertes à l'intérieur de l'environnement de travail.

'Fenêtre' 'Messages'

Raccourci : <Maj>+<Echap>

Cette commande vous permet d'ouvrir ou de fermer la fenêtre de messages avec les messages relatifs aux derniers procédés de compilation, de vérification et de comparaison effectués.

Si la fenêtre de messages est ouverte, un crochet (✓) apparaît devant la commande, dans le menu.

'Fenêtre' 'Gestionnaire de bibliothèques'

Cette commande vous permet d'ouvrir ou de fermer la fenêtre de 'Gestionnaire de bibliothèques'.

'Fenêtre' 'Journal'

Cette commande vous donne accès à la fenêtre Journal qui permet d'afficher des protocoles relatifs aux sessions En ligne. (voir chapitre 6.5, Ressources, Journal).

4.8 Gestionnaire d'Aide

'Aide' 'Sommaire et Index'

Les commandes **Contenu** et **Rechercher** dans le menu **Aide** vous permettent d'ouvrir la fenêtre rubriques d'aide.

Dans l'onglet **Contenu** vous trouvez la table des matières. Les livres s'ouvrent et se ferment au moyen d'un double-clic ou des boutons correspondants. En double-cliquant sur une rubrique marquée ou en activant le bouton **Indiquer**, cette rubrique est affichée dans la fenêtre principale d'aide ou dans la fenêtre mots clé.

Pour chercher un mot clé précis, cliquez sur l'onglet **Index,** et cliquez sur l'onglet **Chercher** pour effectuer une recherche sur l'ensemble du texte. Conformez vous aux instructions dans les onglets.

Fenêtre rubriques d'aide

Aide adaptée au contexte

Raccourci : <F1>

Vous pouvez activer la touche <F1> dans une fenêtre active, une boîte de dialogue ou lorsqu'un élément de menu est sélectionné. Dans le cas des éléments de menu, c'est l'aide qui correspond à la commande actuellement sélectionnée qui est affichée.

Vous pouvez aussi marquer un texte (p.ex. un mot clé ou une fonction standard) et obtenir que l'aide correspondante soit affichée.

5 Les Editeurs

5.1 Pour toutes les éditeurs

Structure d'un éditeur

Tous les éditeurs pour les langages de programmation dans CoDeSys sont constitués d'une partie de déclaration et d'un corps. Le corps peut être composé d'un éditeur graphique ou littéral, la partie de déclaration quant à elle est toujours un éditeur littéral. Le corps et la partie de déclaration sont séparés par une barre horizontale de fractionnement, que l'on peut déplacer à sa convenance, en cliquant dessus avec la souris et en la déplaçant vers le haut et vers le bas tout en maintenant la touche de la souris enfoncée.

Zone d'impression

Les délimitations horizontales et verticales relatives aux pages et en vigueur lors de l'impression du contenu de l'éditeur sont marquées par les lignes rouges pour autant que vous ayez sélectionné l'option '**Zone d'impression**' parmi les options de projet de la boîte de dialogue 'Environnement de travail'. À cet effet, les données par défaut de l'imprimante configurée sont d'application, de même que la mise en page sélectionnée pour l'impression dans le menu 'Fichier' 'Configuration Documentation'. Si aucune imprimante ou mise en page n'est disponible, on se base sur une assignation par défaut (Défaut DFR et Imprimante standard). Les limites horizontales d'impression sont affichées comme si on avait sélectionné l'option 'Nouvelle page pour chaque objet' ou 'Nouvelle page pour chaque sous-objet' dans la 'Configuration Documentation'. La limite inférieure n'est pas représentée.

Veuillez noter : Un affichage précis des zones limites d'impression n'est obtenu qu'avec un facteur zoom réglé à 100%.

Commentaires

Les commentaires de l'utilisateur doivent être enfermés dans les chaînes de caractères spécifiques "(*" et "*)".

Les commentaires sont autorisés dans tous les éditeurs littéraux, et ce à n'importe quel endroit. Autrement dit, les commentaires sont autorisés dans les déclarations, en langage IL et en langage ST, et dans les types de données définis par l'utilisateur. Si le projet est imprimé moyennant l'utilisation d'un **document modèle**, le commentaire saisi en regard de la déclaration de variable apparaît toujours derrière cette variable, ceci pour les parties de programme se basant sur des textes.

Pour chaque réseau, il est possible d'introduire un commentaire dans les éditeurs graphiques en langage FBD et LD Pour cela, choisissez le réseau que vous souhaitez commenter et activez "**Insérer**" + "**Commentaire**". En langage LD, vous pouvez ajouter un commentaire à chaque contact ou chaque bobinage, si le réglage dans l'affichage des options du menu 'Extras' 'Options' le permet. Avec l'éditeur CFC , il existe des modules spéciaux de commentaire qui peuvent être placés selon son gré.

En SFC, vous pouvez entrer des commentaires relatifs aux étapes dans la boîte de dialogue pour l'édition d'attributs d'étape.

Des '**commentaires imbriqués**' sont également permis, pour autant que l'option correspondante ait été activée dans la boîte de dialogue **'Projet' 'Options' 'Options de compilation'**.

Si, dans le mode En Ligne, vous maintenez pendant une courte durée le pointeur de la souris sur une variable, alors le type et, le cas échéant, le commentaire de celle-ci sont visualisés à l'intérieur d'une **info-bulle**.

Zoom sur module appelé

Cette commande est disponible dans le menu contextuel (<F2>) ou dans le menu Extras, pour autant que le curseur se trouve sur le nom du module appelé dans les éditeurs littéraux, ou si la boîte d'un

module est marquée dans les éditeurs graphiques. Le zoom ouvre le module concerné dans sa fenêtre d'éditeur. S'il s'agit d'un module issu d'une bibliothèque, alors le gestionnaire de bibliothèques est appelé et le module correspondant est affiché.

Ouvrir l'instance

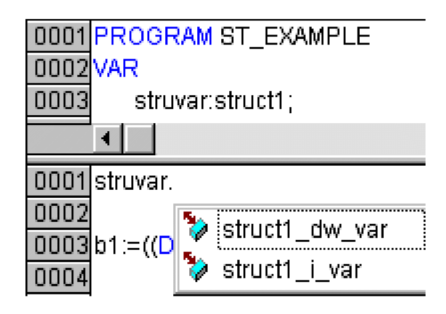
Cette commande correspond à la commande 'Projet' 'Ouvrir l'instance' . Elle est disponible dans le menu contextuel ou dans le menu Extras, pour autant que le curseur se trouve sur le nom du bloc fonctionnel dans les éditeurs littéraux, ou si la boîte d'un module est marquée dans les éditeurs graphiques.

Fonction Intellisense

Si l'option Lister les composants est activée parmi les options de projet dans la catégorie Éditeurs, la fonction Intellisense est disponible dans tous les éditeurs, dans le gestionnaire d'espion et de recettes, dans la visualisation et dans la configuration de l'histogramme.

- Si un point « . » est donné en tant qu'identificateur, une liste de sélection de toutes les variables locales et globales s'ouvre. Vous pouvez sélectionner un élément hors de cette liste et l'insérer en appuyant sur la touche d'entrée après le point. L'insertion se produit également après un double-clic sur l'élément de la liste.
- Si l'identificateur est une instance d'un bloc fonctionnel ou une variable définie comme une structure et suivie d'un point, une liste des variables d'entrée et de sortie du bloc fonctionnel ou des composants de structure s'ouvre après la saisie du point.

  Exemple :
  Saisie de "struvar." -> les composants de la structure struct1 sont affichés :



5.2 L'éditeur de déclaration

5.2.1 Généralités quant aux éditeurs de déclaration

Les éditeurs de déclaration servent à la déclaration de variables pour les modules et les variables globales, à la déclaration de types de données et au niveau du gestionnaire d'espion et de recettes. Ils disposent de toutes les fonctions normales de Windows, et les fonctionnalités IntelliMouse peuvent également être utilisées si le pilote adéquat est installé.

En mode Écrasement, le symbole correspondant est affiché en noir dans la barre des statuts, et vous pouvez commuter entre le mode Écrasement et le mode Insertion en appuyant sur la touche <Inser>.

La déclaration de variables est assistée par la **coloration de la syntaxe**.

Vous trouvez les commandes les plus importantes dans le menu contextuel (touche droite de la souris ou <Ctrl>+<F10>).

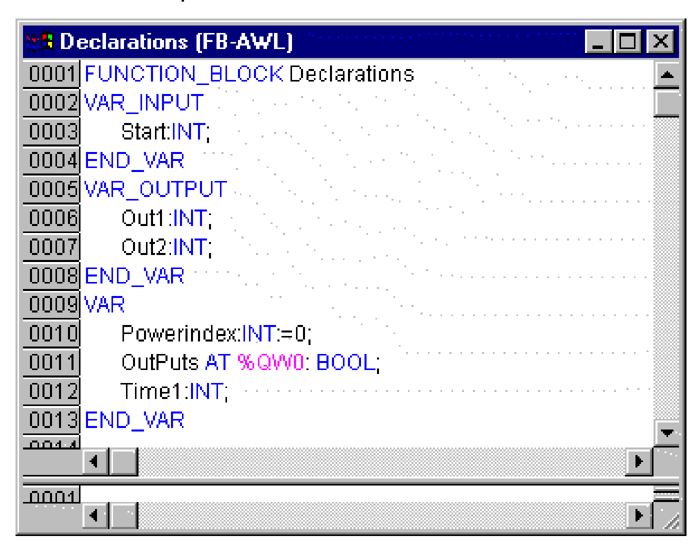
Partie de déclaration

Dans la partie de déclaration d'un module sont déclarées toutes les variables qui sont utilisées exclusivement dans ce module. Il peut s'agir de variables d'entrée, de variables de sortie, de variables

d'entrée-sortie, de variables rémanentes ou de constantes. La syntaxe de déclaration est conforme à la norme IEC61131-3.

Notez que pour le remplissage de la partie déclaration lors de la création d'un nouvel objet de type 'Variables globales', 'Type de fichier', 'Fonction', 'Bloc fonctionnel' ou 'Programme', il est possible d'utiliser des objets modèles, voir chapitre 4.3.

Voici un exemple de déclaration de variables correcte dans l'éditeur **CoDeSys**:



Variables d'entrée

Entre les mots clés **VAR_INPUT** et **END_VAR** sont déclarées toutes les variables qui interviennent en tant que variables d'entrée d'un module, c.-à-d. que la valeur de ces variables peut également être entrée au moment de l'appel, au niveau de l'endroit d'appel.

Exemple :

```
VAR_INPUT
 in1:INT; (* première variable d'entrée *)
END_VAR
```

Voici un exemple d'accès à une variable d'entrée d'un bloc fonctionnel:

Le bloc fonctionnel FUB a une variable d'entréein1 du type INT.

Déclaration:

```
PROGRAM prog
VAR
 inst:FUB;
END_VAR
```

Implémentation en langue IL:

```
LD 17
ST  inst.in1
CAL inst
```

Implémentation en langue ST:

```
inst(in1:=17);
```

Variables de sortie

Entre les mots clés **VAR_OUTPUT** et **END_VAR** sont déclarées toutes les variables qui interviennent comme variables de sortie d'un module, c.-à-d. que les valeurs de ces variables sont renvoyées au module d'appel, pour y être contrôlées et réutilisées.

Exemple :

```
VAR_OUTPUT
 out1:INT; (* première variable de sortie *)
```

```
END_VAR
```

Variables d'entrée-sortie

Entre les mots clés **VAR_IN_OUT** et **END_VAR** sont déclarées toutes les variables qui interviennent comme variables d'entrée-sortie d'un module.

Attention : dans le cas des variables d'entrée-sortie, la valeur de la variable transférée est modifiée directement ("transfert sous forme de pointer", Call-by-Reference). C'est pourquoi une telle variable n'accepte que des constantes comme valeurs d'entrée. Ceci explique également pourquoi les variables d'entrée-sortie VAR_IN_OUT ne peuvent être lues ou écrites directement de l'extérieur par <Instance de bloc fonctionnel> <Variables d'entrée-sortie>.

Exemple :

```
VAR_IN_OUT
 inout1:INT; (* première variable d'entrée-sortie *)
END_VAR
```

Variables locales

Entre les mots clés **VAR** et **END_VAR** sont déclarées toutes les variables locales d'un module. Celles-ci n'ont pas de relation avec l'extérieur, c.-à-d. qu'il est impossible d'y accéder à partir de l'extérieur.

Exemple :

```
VAR
 loc1:INT; (* première variable locale *)
END_VAR
```

Variables rémanentes

Les variables rémanentes gardent leur valeur au-delà de l'exécution normale de programme. Les variables rémanentes ("retain") et les variables persistantes en font partie.

Exemple :

```
VAR RETAIN
 rem1:INT; (* première variable rémanente *)
END_VAR
```

- Les variables rémanentes ("retain") sont caractérisées par le mot clé RETAIN. Ces variables conservent leur valeur même après un arrêt non intentionnel de l'automate ou après une remise à zéro ('En Ligne' 'Reset'). Lors d'un nouveau lancement du programme, le travail continue avec les valeurs mémorisées. Un exemple d'application possible est un compteur installé sur une installation de production, qui doit pouvoir continuer à compter après une panne de secteur. Les variables rémanentes sont réinitialisées à 'En Ligne' 'Reset à froid', 'En Ligne' 'Reset origine' et - au contraire des variables persistantes - , à chaque nouveau téléchargement de programme. Toutes les autres variables sont réinitialisées, soit avec leurs valeurs d'initialisation, soit avec des initialisations standard.
- Les variables persistantes sont caractérisées par le mot clé PERSISTENT. Elles gardent leur valeur seulement après un nouveau téléchargement ('En Ligne' 'Lancer'), mais - au contraire des variables rémanentes ("retain") - non après 'En Ligne' 'Reset', 'En Ligne' 'Reset à froid', 'En Ligne' 'Reset origine', vu qu'elles ne sont pas enregistrées dans la "zone mémoire rémanente". Si les variables persistantes doivent garder leur valeur même après un arrêt non intentionnel de l'automate, elles doivent donc être déclarées en plus comme des VAR RETAIN. Un exemple d'application possible de "variable rémanente persistante" est un compteur pour calculer le nombre d'heures de service effectif, qui doit pouvoir continuer à compter après une panne de secteur.

x = valeur est conservé - = valeur est réinitialisé

apres command 'En Ligne'...	**VAR**	**VAR RETAIN**	**VAR PERSISTENT**	**VAR RETAIN PERSISTENT VAR PERSISTENT RETAIN**
Reset	-	x	-	x
Reset à froid	-	-	-	-
Reset origine	-	-	-	-
Lancer (=Download)	-	-	x	x

Attention :

- Si une variable locale est déclarée dans le programme comme variable rémanente, c'est précisément cette variable qui sera sauvegardée dans la zone mémoire rémanente (comme une variable rémanente globale).
- Si une variable locale est déclarée dans un bloc fonctionnel comme variable rémanente, l'instance complète de ce bloc fonctionnel sera enregistrée dans la zone mémoire rémanente (toutes les données du module), les variables rémanentes déclarées étant cependant seules traitées comme telles.
- Si une variable locale d'une fonction est déclarée comme variable rémanente, ceci n'a pas d'effet. La variable n'est pas enregistrée dans la zone mémoire rémanente ! Si une variable locale d'une fonction est déclarée comme variable persistante, ceci n'a pas d'effet non plus !

Constantes, Typed Literals

Les constantes sont caractérisées par le mot clé **CONSTANT**. Elles peuvent être déclarées de façon locale ou globale. Reportez-vous également à la possibilité d'utilisation de constantes typées (libellés typés - Typed Literals).

Syntaxe:

```
VAR CONSTANT
 <Identificateur>:<Type> := <Initialisation>;
END_VAR
```

Exemple :

```
VAR CONSTANT
 con1:INT:=12; (* première constante *)
END_VAR
```

Vous trouverez au chapitre 9.2.

Variables externes

Les variables globales qui doivent être importées dans le module sont caractérisées par le mot-clé EXTERNAL. Elles apparaissent également en mode En ligne dans la fenêtre d'espion de la partie de déclaration.

Si la déclaration globale ne concorde pas avec la déclaration de variable externe, le message d'erreur suivant s'affiche à l'écran : "Déclaration de '<var>' ne correspond pas à la déclaration globale!!"

Si la variable globale n'existe pas, le message suivant est affiché : "Variable globale inconnue '<var>'!"

Exemple :

```
VAR EXTERNAL
 var_ext1:INT:=12; (* 1. variable externe *)
END_VAR
```

Mots clés

Les mots clés s'écrivent en majuscule dans tous les éditeurs . Il n'est pas permis de choisir des mots clés comme nom de variable.

Déclaration de variables

La syntaxe d'une déclaration de variables se présente comme suit:

<Identificateur> {**AT** <Adresse>}**:**<Type> {**:=** **<**Initialisation>};

Les parties entre accolades {} sont facultatives.

Un identificateur est une série de lettres, chiffres et caractères de soulignement qui commence avec une lettre ou un caractère de soulignement. En ce qui concerne les identificateurs (noms) de variables, il faut veiller à ce qu'ils ne comportent ni espace ni tréma, à ce qu'ils ne soient pas déclarés deux fois et à ce qu'ils ne coïncident pas avec des mots-clés. Les variables peuvent s'écrire indifféremment en majuscules ou en minuscules, c.-à-d. que VAR1, Var1 et var1 représentent une seule et même variable. Les caractères de soulignement sont significants, c.-à-d. que "A_BCD" et "AB_CD" sont interprétés comme deux identificateurs différents. Les caractères de soulignement successifs ne sont autorisés ni au début ni à l'intérieur d'un identificateur. La longueur des identificateurs ainsi que la zone significative sont illimitées.

Toutes les déclarations de variables et tous les éléments appartenant aux types de données peuvent contenir des **initialisations** (attribution d'une valeur initiale). Celles-ci sont constituées d'un opérateur d'affectation " := ". Dans le cas de variables ayant un type élémentaire, les initialisations sont des constantes. L'initialisation par défaut pour toutes les déclarations est 0.

Exemple :

```
var1:INT:=12; (* variable integer avec la valeur initiale 12*)
```

Si vous voulez placer directement une variable à une adresse donnée, vous devez déclarer la variable avec le mot clé **AT**.

Pour entrer plus rapidement les déclarations, vous pouvez utiliser **le mode raccourci**.

Dans des blocs fonctionnels, les variables peuvent être spécifiées même en utilisant des adresses incomplètes. Pour utiliser de telles variables dans une instance locale, vous devez effectuer l'entrée appropriée dans la configuration des variables.

Tenez compte de la possibilité de la **déclaration automatique**.

Déclaration AT

Si vous voulez placer directement une variable à une adresse donnée, vous devez déclarer la variable avec le mot clé **AT**. L'avantage d'un tel procédé réside dans le fait que l'on peut donner à une adresse un nom plus parlant, et aussi dans le fait qu'il suffit d'effectuer à un seul endroit (plus précisément au niveau de la déclaration) une éventuelle modification au niveau d'un signal d'entrée ou d'un signal de sortie.

Veillez à ce que l'on ne puisse pas avoir un accès en écriture à des variables qui sont placées sur une entrée. Une autre contrainte réside dans le fait que les déclarations AT peuvent être effectuées uniquement pour des variables locales ou globales, et non pour des variables d'entrée ou de sortie de module.

Veillez à ce que l'on ne puisse pas avoir un accès en écriture à des variables placées sur une entrée.

Exemples:

```
commutateur_chauffage7 AT %QX0.0: BOOL;
impulsion_barrage_photoelectrique AT %IX7.2: BOOL;
récepteur AT %MX2.2: BOOL;
```

Remarque : Lorsque des variables booléennes sont alignées sur une adresse Byte, Word ou DWORD, elles occupent 1 octet entier avec TRUE ou FALSE et pas seulement le premier bit après l'offset !

'Insérer' 'Mots clés de déclaration'

Cette commande entraîne l'ouverture d'une liste reprenant tous les mots clés pouvant être utilisés dans la partie de déclaration d'un module. Après qu'un mot clé a été sélectionné et que ce choix a été confirmé, le mot est inséré à l'endroit désigné par la position actuelle du curseur.

Vous pouvez aussi obtenir cette liste en appelant la liste de sélection (<F2>) pour l'édition et en sélectionnant la catégorie **Déclarations**.

'Insérer' 'Types'

Cette commande vous présente les types qui peuvent être utilisés pour une déclaration de variables. Vous pouvez aussi obtenir cette liste en appelant la liste de sélection pour l'édition.

Les types sont répartis en catégories:

- Types standard BOOL, BYTE etc.
- Types définis Structures, types d'énumération etc.
- Blocs fonctionnels standard pour des déclarations d'instance
- Blocs fonctionnels définis pour des déclarations d'instance

CoDeSys supporte tous les types standard de la norme IEC61131-3:

Coloration de la syntaxe

Vous recevez une aide visuelle dans l'implémentation et la déclaration de variables, dans les éditeurs littéraux et dans l'éditeur de déclaration. Le fait que le texte soit visualisé en couleurs permet d'éviter des erreurs ou de les détecter plus rapidement.

Un commentaire non fermé, qui termine un ensemble de commentaires relatif à des instructions, est repéré immédiatement. Les mots clés ne comportent pas d'erreurs involontaires etc.

Les conventions de couleurs sont les suivantes:

- Bleu Mots clés
- Vert Commentaires
- Rose Valeurs de type booléen (TRUE/FALSE)
- Rouge Entrée erronée (p.ex. constante de temps non valable, mot clé écrit en minuscule, …)
- Noir Variables, constantes, opérateurs d'affectation,…

Mode raccourci

L'éditeur de déclaration de **CoDeSys** vous offre la possibilité d'utiliser le mode raccourci. Celui-ci est activé lorsque vous terminez une ligne par <Ctrl>+<Entrée>.

Les raccourcis suivant sont supportés:

- Tous les identificateurs dans une ligne, à l'exception du dernier, sont changés en identificateurs des variables intervenant dans la déclaration.
- Le type de la déclaration est fixé par le dernier identificateur de la ligne:

  B ou BOOL fournit comme résultat BOOL
  I ou INT fournit comme résultat INT
  R ou REAL fournit comme résultat REAL
  S ou STRING fournit comme résultat STRING

- Si ces règles ne suffisent pas à définir un type, alors c'est le type BOOL qui est retenu, et le dernier identificateur n'est pas utilisé comme type (exemple 1).
- Chaque constante est changée en une initialisation ou en une longueur de chaîne de caractères, en fonction du type de déclaration (exemples 2 et 3).
- L'attribut AT … est ajouté à une adresse (comme dans %MD12) (exemple 4).
- Un texte après un point-virgule (;) est changé en commentaire (exemple 4).
- Tous les autres caractères de la ligne sont ignorés (comme par exemple le point d'exclamation dans l'exemple 5).

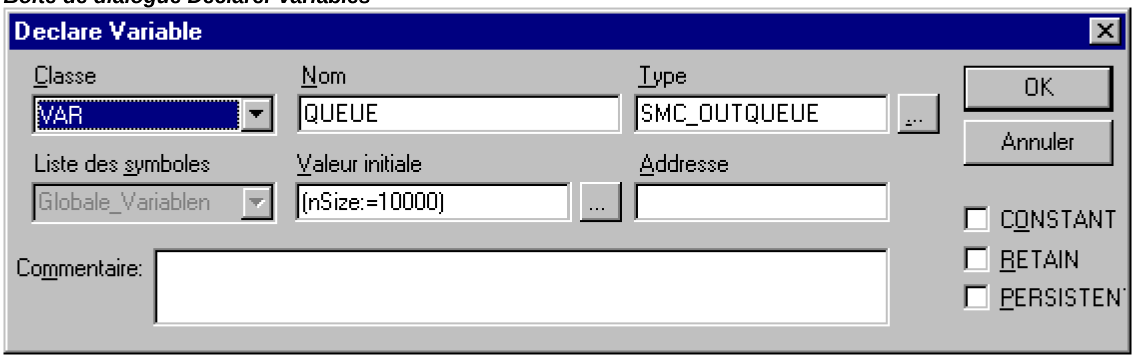
Exemples:

Raccourci	Déclaration
`A`	`A: BOOL;`
`A B I 2`	`A, B: INT := 2;`
`ST S 2; une chaîne de caractères`	`ST: STRING(2); (* une chaîne *)`
`X %MD12 R 5; nombre réel`	`X AT %MD12: REAL := 5.0;(* nombre réel *)`
`B !`	`B: BOOL;`

Déclarer automatiquement

Si l'option **Déclarer automatiquement** a été sélectionnée, alors apparaît dans chaque éditeur, après entrée d'une variable non encore déclarée, une boîte de dialogue qui aide à déclarer cette variable.

Boîte de dialogue Déclarer variables



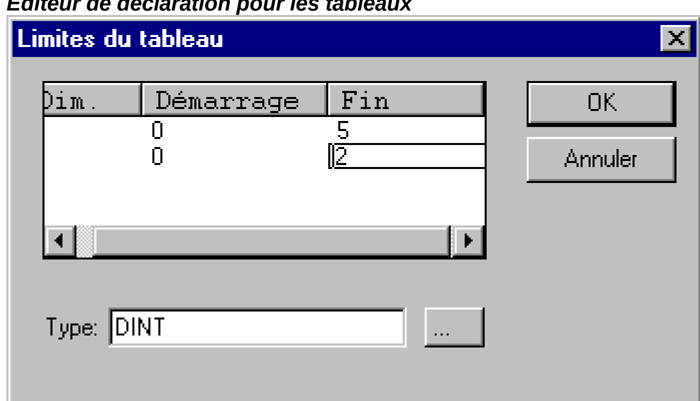
Dans la zone de liste modifiable **Classe**, opérez une sélection pour déclarer cette variable, soit comme variable locale (**VAR**), soit comme variable d'entrée (**VAR_INPUT**), soit comme variable de sortie (**VAR_OUTPUT**), soit comme variable d'entrée-sortie (**VAR_INOUT**), soit comme variable globale (**VAR_GLOBAL**).

Les options **CONSTANT** et **RETAIN, PERSISTENT** vous permettent de définir s'il s'agit d'une constante ou d'une variable rémanente.

Le nom de variable entré dans l'éditeur a été préaffecté au champ **Nom**, tandis que BOOL a été préaffecté au champ **Type**. Le bouton **...** vous permet d'accéder à la boîte de dialogue de la liste de sélection pour l'édition, afin d'opérer une sélection parmi tous les types possibles.

Si vous sélectionnez comme type de variable « ARRAY » (tableau), la boîte de dialogue permettant la saisie des 'Limites du tableau' s'affiche' :

Éditeur de déclaration pour les tableaux



Pour chacune des trois dimensions (**Dim.**), vous pouvez saisir les limites du tableau sous **Démarrage** et **Fin** en cliquant sur le champ correspondant. Le type de tableau est donné dans le champ **Type**. Le bouton vous permet d'appeler une boîte de dialogue Liste de sélection pour l'édition.

En quittant la boîte de dialogue des limites du tableau via le bouton **OK**, les données du champ 'Type' dans la boîte de dialogue de 'Déclaration de variable' sont remplies dans le format CEI. Exemple : ARRAY [1..5, 1..3] OF INT

Vous pouvez entrer la valeur initiale de la variable à déclarer dans le champ **Valeur initiale**. Si celle-ci est un tableau ou une structure valable, vous pouvez ouvrir une boîte de dialogue spéciale pour l'initialisation par le biais du bouton ou , ou ouvrir une boîte de dialogue Liste de sélection pour l'édition pour tous les autres types.

Dans la boîte de dialogue d'initialisation pour un tableau, vous obtenez une liste de tous les éléments de tableau et vous pouvez ouvrir, à l'aide d'un clic de la souris derrière le ":=", un champ d'édition pour la saisie de la valeur initiale d'un élément.

Dans la boîte de dialogue d'initialisation pour une structure, les composants individuels sont présentés sous la forme d'une arborescence. Le type et la valeur initiale par défaut sont donnés entre parenthèses derrière le nom de la variable, suivi eux aussi de ":=". En cliquant avec la souris sur le champ derrière ce ":=", vous ouvrez un champ d'édition dans lequel vous pouvez saisir la valeur initiale souhaitée. Si un des composants est un tableau, vous pouvez, à l'aide d'un clic de la souris sur le signe Plus devant le nom du tableau, faire apparaître les champs individuels du tableau et les éditer avec les valeurs initiales.

En quittant la boîte de dialogue avec **OK**, l'initialisation du tableau ou de la structure en format CEI apparaît dans le champ **Valeur initiale** de la boîte de dialogue de déclaration.

Exemple : x:=5,champ:=2,3,struct2:=(a:=2,b:=3)

Dans le champ **Adresse** vous pouvez placer une variable à déclarer sur une adresse CEI (déclaration AT).

Le cas échéant, entrez un **Commentaire**. Il est possible d'insérer des retours à la ligne dans le commentaire à l'aide des touches "Ctrl" + "Entrée".

Si vous appuyez sur **OK**, la boîte de dialogue de déclaration se referme et la variable convertie en syntaxe CEI est entrée dans l'éditeur de déclaration approprié.

Remarque : La boîte de dialogue de déclaration de variables peut également être ouverte en cas de besoin en utilisant la commande "Editer" + "Déclarer automatiquement".

Si le curseur se trouve sur une variable, la fenêtre de déclaration automatique ainsi que les paramètres relatifs aux variables peuvent être ouverts en mode Hors ligne via les touches <Maj> <F2>.

Numéros de ligne dans l'éditeur de déclaration

En mode hors ligne, un simple clic sur un numéro de ligne donné suffit pour marquer toute la ligne de texte correspondante.

Dans le mode En Ligne, un simple clic sur un numéro de ligne donné permet d'afficher ou de masquer la variable qu'elle contient, dans le cas où il s'agit d'une variable structurée.

Déclarations sous forme de tableau

Si l'option **Déclarations sous forme de tableau** a été configurée au niveau de la catégorie **Editeur,** dans la boîte de dialogue Options, l'éditeur de déclaration est proposé sous forme de tableau. Comme pour une cartothèque, vous pouvez sélectionner séparément les onglets correspondant aux différents types de variables et éditer les variables.

Pour chaque variable les champs suivants vous sont proposés, afin d'effectuer une saisie:

Nom: Entrez l'identificateur de la variable.

Adresse: Le cas échéant, entrez l'adresse de la variable (déclaration AT).

Type: Entrez le type de la variable. (lors de la création d'une instance de bloc fonctionnel, entrez le bloc fonctionnel)

Valeur initiale: Le cas échéant, entrez l'initialisation de la variable (correspondant à l'opérateur d'affectation " := ").

Commentaire: Entrez un commentaire à cet endroit.

Il est possible de basculer sans aucune difficulté d'un type de visualisation à l'autre. Dans le mode En Ligne, il n'existe pas de différences au niveau de la visualisation de l'éditeur de déclaration.

Utilisez la commande 'Insérer' 'Nouvelle déclaration' pour éditer une nouvelle variable.

Editeur de déclaration sous forme de tableau

PLC_PRG (PRG-AS)

VAR | VAR_INPUT | VAR_OUTPUT | VAR_IN_OUT | CONS

	Name	Adresse	Typ	Initial	Komm
0001	AMPEL1		AMPEL		
0002	AMPEL2		AMPEL		
0003	VERZ		WARTEN		
0004	ZAEHLER		INT		

'Insérer' 'Nouvelle déclaration'

Cette commande vous permet de saisir une nouvelle variable dans le tableau de déclaration de l'éditeur de déclaration. Si le curseur est actuellement positionné dans un champ du tableau, la nouvelle variable est insérée devant cette ligne, sinon elle est insérée à la fin du tableau. En outre, vous pouvez insérer à la fin du tableau une nouvelle déclaration, en actionnant dans le dernier champ du tableau la touche directionnelle de droite ou la touche du tabulateur.

Vous obtenez une variable pour laquelle le texte 'Nom' a été prédéfini dans le champ **Nom** et pour laquelle le texte 'Bool' a été prédéfini dans le champ **Type**. Il convient que vous remplaciez ces valeurs par les valeurs de votre choix. Le nom et le type sont suffisants pour une déclaration de variable complète.

5.2.2 Editeurs de déclaration dans le mode En Ligne

Dans le mode En Ligne le déclarateur d'édition est changé en fenêtre d'espion. Chaque ligne comprend une variable, qui est suivie d'un caractère égal (=) et de la valeur de la variable. Si la variable n'a pas encore été définie, trois points d'interrogation (???) apparaissent. Dans le cas des blocs fonctionnels, la valeur n'est affichée que pour les instances ouvertes (commande 'Projet' 'Ouvrir l'instance').

Chaque variable constituée de plusieurs éléments est précédée du signe plus. En appuyant sur la touche <Entrée> ou en double-cliquant sur une telle variable, celle-ci est affichée. Dans l'exemple, c'est la structure Signal1 qui serait affichée.

```
⊟ AMPEL1
   .STATUS = 3
   .GRUEN = FALSE
   .GELB = FALSE
   .ROT = TRUE
   .AUS = FALSE
```

Dans le cas où une variable est affichée, tous ses composants sont affichés l'un après l'autre. La variable est précédée d'un caractère moins. En double-cliquant une nouvelle fois ou en appuyant sur la touche <Entrée> les éléments de la variable sont masqués et le signe plus apparaît à nouveau.

En appuyant sur la touche <Entrée> ou en double-cliquant sur une variable à un seul élément, la boîte de dialogue pour écrire les variables s'ouvre. Celle-ci permet de modifier la valeur actuelle de la variable. Dans le cas des variables booléennes il n'apparaît pas de boîte de dialogue; la valeur change directement d'un état à l'autre.

La nouvelle valeur est affichée couleur turquoise derrière la variable entre les signes '<' et '>' et reste inchangée.

En donnant la commande 'En Ligne' 'Ecrire valeurs des variables', les valeurs sélectionnées sont affectées à toutes les variables et ces dernières se colorent à nouveau en noir.

En donnant la commande 'En Ligne' 'Forcer valeurs des variables', les valeurs sélectionnées sont affectées à toutes les variables, jusqu'à ce que la commande 'Arrêter de forcer' soit effectuée. Dans ce cas, la couleur de la valeur de forçage vire au rouge.

5.2.3 Instructions Pragma

Instruction Pragma

L'instruction Pragma sert au contrôle du processus de compilation. Elle est donnée dans une ligne de programme comme texte supplémentaire ou encore dans sa propre ligne d'éditeur de déclaration.

L'instruction Pragma est donnée entre accolades (majuscules ou minuscules n'ont aucune importance) :

{ <texte de l'instruction> }

Si le compilateur ne parvient pas à interpréter le texte de l'instruction, la Pragma entière est considérée comme étant un commentaire et est ignorée. Un message vous est cependant donné : "Ignore directive du compilateur "<texte de l'instruction>"!".

Selon le type et le contenu de la Pragma, celle-ci agit sur la ligne dans laquelle elle se trouve ou sur les lignes suivantes et ce jusqu'à ce qu'elle soit levée, que la même pragma soit exécutée avec d'autres paramètres, ou qu'on ait atteint la fin du fichier. On comprend ici sous le terme fichier : partie de déclaration, partie d'implémentation, liste de variables globale, déclaration de types.

L'accolade d'ouverture peut suivre le nom d'une variable. Les accolades d'ouverture et de fermeture doivent se trouver sur la même ligne.

À l'heure actuelle, les pragmas ci-dessous peuvent être utilisées dans CoDeSys :

- Pragma {flag} pour l'initialisation, l'espionnage et la création d'icônes
- Pragma {bitaccess...} pour accès binaire
- Pragma {parameter..}, {template...}, {instance...} pour la création d'entrées dans le gestionnaire des paramètres

La Pragma suivante est actuellement à disposition :

Pragmas pour la création d'entrées dans le gestionnaire des paramètres

Grâce aux pragmas au sein des déclarations de variables, il est possible de créer automatiquement, dans la liste des paramètres, des entrées gérées par le biais du gestionnaire des paramètres. Le Gestionnaire des paramètres est selon le système cible disponible dans le système de programmation, ce qui signifie qu'il doit être activé dans la configuration du système cible (fonctions réseau).

Généralités sur la syntaxe :

- Une pragma est donnée entre des accolades. L'écriture en majuscules ou en minuscules n'a aucune importance. Si la pragma est ajoutée à des déclarations de variables "normales", elle doit

l'être juste avant le point-virgule de clôture de la déclaration de variable sur laquelle elle est sensée agir.

- Les pragmas utilisées dans des fenêtres VAR_CONFIG sont données chacune dans une seule ligne et ne sont pas clôturées par un point-virgule !
- <name> (=nom): nom de la liste de paramètres dans le gestionnaire des paramètres. Si la liste de variables n'existe pas encore, elle est automatiquement créée.
- <key> (=clé): nom de l'attribut, c.-à-d. titre de colonne dans la liste des paramètres ; par exemple "nom", "valeur", "niveau d'accès" etc.; les clés pouvant être indiquées sont fonction de la définition du type de liste de paramètres.
- <value> (=valeur): valeur selon laquelle l'attribut défini par <clé> doit être entré dans la liste. Veuillez noter à cet égard que les valeurs qui contiennent des espaces doivent être données entre guillemets doubles. Exemple : ...`niveau d'accès="read only"`...
- Toutes les définitions de clés dans les pragmas sont données l'une à la suite de l'autre dans des crochets, séparées par des espaces.

Remarque: Les énoncés de pragmas agissent déjà dès que l'on passe à autre chose (précompilation), et donc dès que l'on quitte l'éditeur de déclaration. Les entrées de pragmas incorrectes ne sont signalées qu'à la compilation du projet.

Les entrées ci-dessous peuvent être créées :

1.**Entrées dans des listes de paramètres de type 'Liste de variables'**

(a) à partir de la partie Déclaration des programmes et des listes de variables globales

Pour une variable au sein d'une déclaration de PROGRAMME ou de VAR_GLOBAL, une entrée peut être créée dans une liste de paramètres de type 'Variables', pour autant qu'elle soit déclarée comme suit: (Si la liste de paramètres n'existe pas encore, elle est créée.)

Syntaxe: {parameter list=<name> [<key>=<value> <key>=<value> ...autres clés] }

Exemple : La variable bvar est déclarée dans un programme, et elle doit être entrée dans la liste de paramètres parlist1, de type liste de variables, avec comme nom bvar1, comme valeur 102, comme index 16#1200 et comme sous-index 16#2.

```
VAR
bvar:INT{parameter list=parlist1 [name=bvar1 value=102 index=16#1200
subindex=16#1                    ]                     };
END_VAR
```

(b) au moyen d'une déclaration dans l'interface VAR_CONFIG:

Pour des variables, une entrée peut être créée dans une liste de paramètres de type 'Variables' si elle est déclarée comme suit dans une fenêtre VAR_CONFIG.(Si la liste de paramètres n'existe pas encore, elle est créée.)

Syntaxe: {parameter list=<name> path=<path> [<key>=<value> <key>=<value> ...autres clés] }

<path> chemin d'accès de la variable selon laquelle l'entrée doit être créée, p.ex. "PLC_PRG.act1.var_x"

Exemple : pour la variable var_x, une entrée est créée dans la liste des paramètres varlist1, et xvar est saisi comme nom symbolique.

```
VAR_CONFIG
{parameter list=varlist1 path=PLC_PRG.act1.var_x [ name=xvar ] }
END_VAR
```

2. Entrées dans des listes de paramètres de type 'Modèle' à partir de blocs fonctionnels et de structures

Avec des déclarations de variables dans des blocs fonctionnels ou des structures, il est possible de créer des entrées dans des listes de paramètres de type 'Modèle'. (Si le modèle n'existe pas encore, il est créé.)

Syntaxe: {template list=<name> [<key>=<value> <key>=<value> ...autres clés] }

Exemple : La variable strvar, élément de la structure stru1, doit être entrée dans le gestionnaire des paramètres sous le nom (member) "struvar1" et avec niveau d‘accès=low dans le modèle "vorl1":

```
TYPE stru :
STRUCT
ivar:INT;
strvar:STRING{template list=vorl1 [member=struvar1 accesslevel=low]
};
END_STRUCT
END_TYPE
```

3. Entrées dans des listes de paramètres de type 'Instance' (pour des variables d'array, de bloc fonctionnel ou structurées)

(a) À partir de programmes et de listes de variables globales

On peut créer directement une liste d'instances dans le gestionnaire des paramètres lors de la déclaration de variables d'array, de bloc fonctionnel ou de structure au sein d'un programme ou d'une liste de variables globales.

Syntaxe: {instance list=<name> template=<template> baseindex=<index> basesubindex=<subindex> [<key>=<value> <key>=<value> ...autres clés] }

Attention : Pour des variables d'array, ne définissez pas de valeur pour la clé "Member", car celle-ci est automatiquement créée à partir de l'index d'array pour chaque élément d'array.

Exemples :

Exemple1: Une variable d'array arr_1 est déclarée comme suit, afin qu'un liste d'instances "arrinst" soit créée dans le gestionnaire des paramètres (si elle n'est toutefois pas encore disponible) dans laquelle les éléments d'array doivent être entrés; chaque élément doit tout d'abord être entré sous le nom symbolique xname (éditable par après au sein de la liste), et le sous-index doit être incrémenté de 1, en partant de 0 (sous-index de base), pour chaque nouvelle entrée.

```
arr_1: ARRAY [1..8] OF INT{instance list=arrinst template=ARRAY
baseindex=16#0 basesubindex=16#0 [name=xname ] };
```

Pour l'exemple1, insérer array dans la liste d'instances

arrinst

Name	Member	Value	Index	SubIn...	Acces...	Accessright	Min	Max
xname	[1]		16#0	16#0	low	read only		
xname	[2]		16#0	16#1	low	read only		
xname	[3]		16#0	16#2	low	read only		
xname	[4]		16#0	16#3	low	read only		
xname	[5]		16#0	16#4	low	read only		
xname	[6]		16#0	16#5	low	read only		
xname	[7]		16#0	16#6	low	read only		
xname	[8]		16#0	16#7	low	read only		

Synchrone Aktionen ☑ Vorlage ARRAY Basisindex 16#0

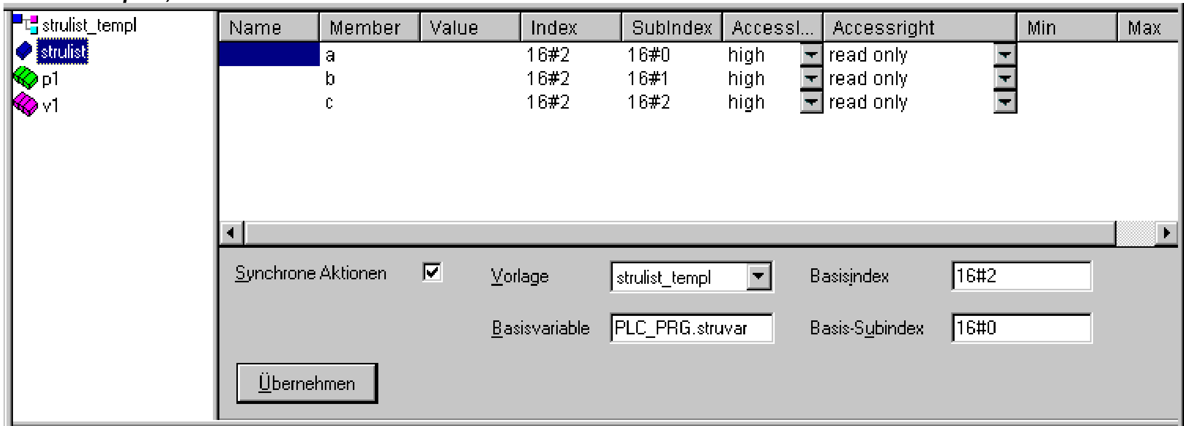
Basisvariable PLC_PRG.arr_1 Basis-Subindex 16#0

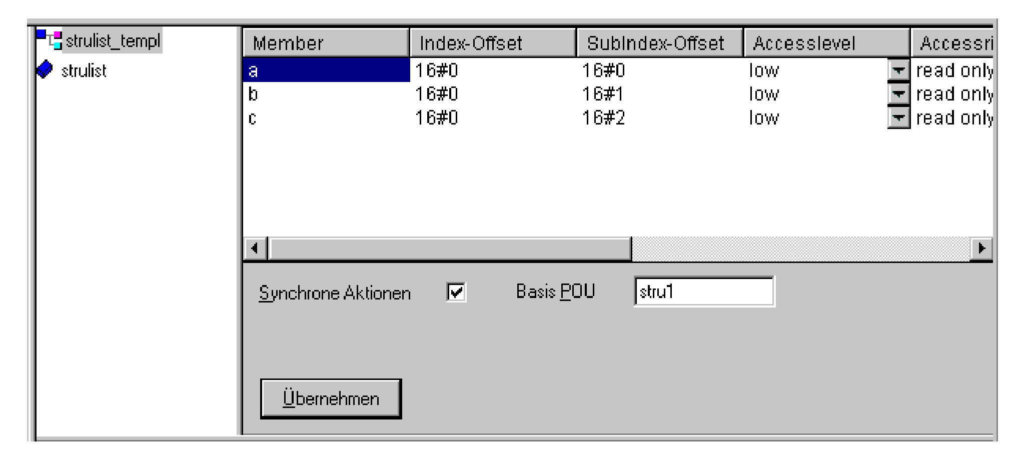
Übernehmen

Exemple2 : Une variable structurée struvar de type stru1 est déclarée comme suit, afin qu'une liste d'instances "strulist" soit créée dans le gestionnaire des paramètres (si elle n'est toutefois pas encore disponible), cette liste se basant sur le modèle strulist_temp et contenant comme entrées les variables a,b,c de la structure existante stru1. Les variables ne reçoivent pas encore de nom symbolique à l'insertion, le niveau d'accès est mis sur High, et l'index tel que défini par le modèle est à chaque fois incrémenté de 2. Veillez à ce que le modèle d'instance indiqué strulist_temp soit présent dans le gestionnaire des paramètres:

```
struvar:stru1{instance list=strulist template=strulist_templ
baseindex=16#2 basesubindex=16#0 [accesslevel=high] };
```

Pour l'exemple2, variable structurée insérée dans la liste d'instances





(b) au moyen d'une déclaration dans l'interface VAR_CONFIG

Pour des variables sujettes à instanciation, des entrées peuvent être directement définies dans une liste d'instances au sein du gestionnaire des paramètres, cela par le biais d'une déclaration dans une fenêtre VAR_CONFIG. Cette déclaration n'est en rien liée à d'éventuelles configurations de variables ! (Si la liste d'instances n'existe pas encore, elle est créée.)

Veillez à ce que le modèle indiqué (template) soit présent dans le gestionnaire des paramètres.

Syntaxe: {instance list=<name> path=<path> template=<template> baseindex=<index> basesubindex=<subindex>[<key>=<value> <key>=<value> ...autres clés] }

<path>: le chemin d'instance de la variable; p.ex. "PLC_PRG.fb1inst", dans lequel fb1inst est une instance du bloc fonctionnel fb1.

Exemple : Grâce à l'entrée ci-dessous dans une fenêtre VAR_CONFIG (indépendamment d'éventuelles configurations de variables!), des entrées sont créées dans une liste d'instances "varinst1" pour toutes les variables du bloc fonctionnel fb1, sur base du modèle "fb1_templ" (obligatoirement disponible). Pour chaque entrée, le décalage d'index tel que prédéfini par le modèle est incrémenté de 2 (index de base), tandis que le décalage de sous-index reste tel quel (sous-index de base). Chaque entrée se voit attribuer un nom symbolique "fb1var" qui peut encore être édité dans la liste.

```
VAR_CONFIG
{instance  list=varinst1  path=PLC_PRG.fb1  template=fb1_templ  baseindex=16#2
           basesubindex=16#0 [ name=fb1var ]}
END_VAR
```

Pragma {flag} pour l'initialisation, l'espionnage et la création de symboles

Syntax: {flag [<flags>] [off|on]}

Cette pragma peut influer les propriétés d'une déclaration d'une variable:

<flags> peut être une combinaison du flags figurantes ci dessous:

noinit:	La variable n'est pas initialisée.
nowatch:	La variable n'est pas gardée.
noread:	La variable est exporte sans d'un droit d'accès lecture dans le fichier du symboles.
nowrite:	La variable est exporte sans d'un droit d'accès écriture dans le fichier du symboles.
noread, nowrite:	La variable n'est pas exporte dans le fichier du symboles.

Avec la modification "**on**" la pragma fait effet sur toutes les déclarations subordonnées, jusqu'elle est enlevée par la pragma {flag **off**}, respectivement jusqu'elle est surimprimée par une autre pragma {flag <flags> on}.

Sans la modification "on" ou "off" la pragma seulement fait effet sur la déclaration actuelle (c'est la déclaration qu'est terminée par le point-virgule prochain).

Exemples par utilisation de la pragma {flag}:

Initialisation et espionnage de variables :

Variable "a" n'est pas initialisée et n'est pas observée. Variable b n'est pas initialisée:

```
VAR
 a : INT {flag noinit, nowatch};
 b : INT {flag noinit };
END_VAR
VAR
 {flag noinit, nowatch on}
 a : INT;
 {flag noinit on}
 b : INT;
 {flag off}
END_VAR
```

Les deux variables ne sont pas initialisées:

```
{flag noinit on}
VAR
a : INT;
b : INT;
END_VAR
{flag off}
VAR
{flag noinit on}
a : INT;
b : INT;
{flag off}
END_VAR
```

Exportation de variables dans le fichier des symboles :

Les drapeaux "noread" et "nowrite" permettent, au sein d'un module disposant du droit de lecture ou d'écriture, de pourvoir les variables individuelles d'un droit d'accès limité.

La configuration par défaut d'une variable est celle relative au module dans lequel cette variable est déclarée. Une variable ne disposant ni d'un droit de lecture, ni d'un droit d'écriture, ne sera pas exportée dans le fichier des symboles.

Exemples:

Le module est pourvu d'un droit de lecture et d'écriture ; les Pragmas suivantes permettent l'exportation de la variable a uniquement avec droit d'accès écriture, et empêchent l'exportation de la variable b:

```
VAR
 a : INT {flag noread};
 b : INT {flag noread, nowrite};
END_VAR

VAR
 { flag noread on}
 a : INT;
 { flag noread, nowrite on}
 b : INT;
 {flag off}
END_VAR
```

Les deux variables ne sont pas exportées dans le fichier des symboles:

```
{ flag noread, nowrite on }
VAR
 a : INT;
 b : INT;
END_VAR
{flag off}

VAR
 { flag noread, nowrite on }
 a : INT;
 b : INT;
 {flag off}
END_VAR
```

La pragma agit de manière additionnelle sur toutes les déclarations subordonnées de variables.

Exemple: (toutes les modules utilisées sont exportées avec droit d'accès lecture et écriture)

```
a : afb;

...

FUNCTION_BLOCK afB
VAR
 b : bfb {flag nowrite};
 c : INT;
END_VAR

...

FUNCTION_BLOCK bfB
VAR
 d : INT {flag noread};
 e : INT {flag nowrite};
END_VAR
```

"a.b.d": N'est pas exportée.

"a.b.e": Exportée seulement avec droit d'accès lecture.

"a.c": Exportée avec droit d'accès lecture et écriture.

Pragma {bitaccess...} pour l'accès binaire

Cette pragma vous permet de définir des accès binaires valides et symboliques à des **structures**, ces accès se faisant à l'aide d'une constante globale. Ces symboles seront alors proposés dans la liste de sélection pour l'édition et la fonction Intellisense et utilisés pour la représentation de l'accès binaire lors de l'espionnage dans l'éditeur de déclaration. Les constantes globales utilisées y sont également affichées.

Veuillez noter: L'option de projet 'Remplacer constantes' (catégorie Options de compilation) doit être activée !

Dans la définition de la structure, la pragma doit être insérée dans une ligne séparée. Cette ligne n'est pas clôturée par un point virgule.

Syntaxe: {bitaccess <Constante globale> <Numéro de bit> '<Commentaire>'}

<Constante globale> : nom de la constante globale qui doit être définie au sein d'une liste de variables globales.

<Numéro de bit> : valeur de la constante globale telle que définie dans la liste des variables constantes.

Pour voir un exemple, reportez-vous au chapitre Appendice B, opérandes dans CoDeSys, adressage de bits au sein de variables.

5.3 Les éditeurs littéraux

5.3.1 Généralités à propos des éditeurs littéraux

Les éditeurs littéraux (l'éditeur de la liste d'instructions et l'éditeur du littéral structuré) de CoDeSys utilisés pour la partie implémentation d'un projet, disposent des fonctionnalités courantes des éditeurs littéraux Windows.

Dans les éditeurs littéraux, l'implémentation est assistée par la coloration de la syntaxe.

Lorsque vous travaillez en mode Refrappe, l'abréviation **'RFP'** est affichée en noir dans la barre d'état. Vous pouvez commuter entre le mode Refrappe et le mode Insertion en actionnant la touche <Inser>.

Vous trouvez les commandes les plus importantes dans le menu contextuel (touche droite de la souris ou <Ctrl>+<F10>).

Les éditeurs littéraux utilisent de façon spécifique les commandes de menu suivantes:

'Insérer' 'Opérateur'

Cette commande entraîne l'affichage dans une boîte de dialogue de tous les opérateurs disponibles dans le langage actuel.

Si un opérateur est sélectionné et si la liste est fermée au moyen de **OK**, alors l'opérateur sélectionné est inséré au niveau de la position actuelle du curseur.

(L'utilisation a lieu de la même façon que dans la Liste de sélection pour l'édition).

'Insérer' 'Opérande'

Cette commande entraîne l'affichage dans une boîte de dialogue de toutes les variables. Vous pouvez opter pour la visualisation de la liste des variables globales, des variables locales ou des variables système.

Si un des opérandes est sélectionné et si la boîte de dialogue est fermée au moyen de **OK**, alors l'opérande sélectionné est inséré au niveau de la position actuelle du curseur.

(L'utilisation a lieu de la même façon que dans Liste de sélection pour l'édition).

'Insérer' 'Fonction'

Cette commande entraîne l'affichage, dans une boîte de dialogue, de toutes les fonctions. Vous pouvez opter pour la visualisation de la liste des fonctions définies par l'utilisateur ou de la liste des fonctions standard.

Si une des fonctions est sélectionnée et si la boîte de dialogue est fermée au moyen de **OK**, alors la fonction sélectionnée est insérée au niveau de la position actuelle du curseur. (L'utilisation a lieu de la même façon que dans la Liste de sélection pour l'édition).

Si l'option **Avec arguments** a été sélectionnée dans la boîte de dialogue, alors les variables d'entrées et de sorties requises par la fonction sont insérées avec celle-ci.

'Insérer' 'Bloc fonctionnel'

Cette commande entraîne l'affichage, dans une boîte de dialogue, de tous les blocs fonctionnels. Vous pouvez optez pour la visualisation de la liste des blocs fonctionnels définis par l'utilisateur ou de la liste des blocs fonctionnels standard.

Si un des blocs fonctionnels est sélectionné, et si la boîte de dialogue est fermée au moyen de **OK**, alors le bloc fonctionnel sélectionné est inséré au niveau de la position actuelle du curseur. (L'utilisation a lieu de la même façon que dans la Liste de sélection pour l'édition).

Si l'option **Avec arguments** a été sélectionnée dans la boîte de dialogue, alors les variables d'entrée et de sortie requises par le bloc fonctionnel sont insérées avec celui-ci. Ceux-ci ne doivent cependant pas obligatoirement être affectés.

Appel de module avec paramètres de sortie

Les paramètres de sortie d'un module appelé peuvent directement être affectés lors de l'appel dans le cas des langages textuels IL ou ST. Exemple : le paramètre de sortie out1 de afbinst est attribué à la variable a.

```
Langage IL :  CAL afbinst(in1:=1, out1=>a)
Langage ST  afbinst(in1:=1, out1=>a);
```

Si le module est inséré dans la fenêtre d'implémentation d'un module ST ou IL, en utilisant la liste de sélection pour l'édition (<F2>) et l'option **Avec arguments**, ce module sera automatiquement représenté avec ses paramètres dans cette syntaxe. Il n'est pas absolument nécessaire d'affecter les paramètres.

Les éditeurs littéraux en mode En Ligne

Les fonctions En Ligne des éditeurs sont Définir point d'arrêt et Pas à pas. Si l'on ajoute à cela l'espionnage, l'utilisateur dispose alors d'une fonctionnalité de débogage comparable à celle existant pour un langage de programmation évolué sous Windows.

Dans le mode En Ligne, la fenêtre de l'éditeur littéral est scindée verticalement en deux parties. Dans la partie gauche de la fenêtre est située le texte de programmation normal et dans la partie droite sont visualisées les variables, dont les valeurs sont modifiées dans la ligne correspondante.

La visualisation est la même que pour la partie de déclaration. Autrement dit, lorsque l'automate programmable est en service, les valeurs instantanées des variables respectives sont visualisées. Les valeurs structurées (tableaux, structures ou instances de bloc fonctionnel) sont signalées par un signe plus placé devant l'identificateur.

En cliquant avec la souris sur le signe plus ou en appuyant sur la touche <Entrée>, la variable est affichée ou masquée.

Lors de l’espionnage d’expressions ou de position binaires d'une variable, veuillez noter : Dans le cas des expressions, la valeur de l’expression entière est affichée. Exemple : a AND b s’affiche en bleu ou avec ":=TRUE" si a et b ont la valeur TRUE). Dans le cas de positions binaires d'une variable, la position binaire concernée est espionnée (p.ex. a.3 sera affiché en bleu ou représenté avec :=TRUE si a la valeur 4).

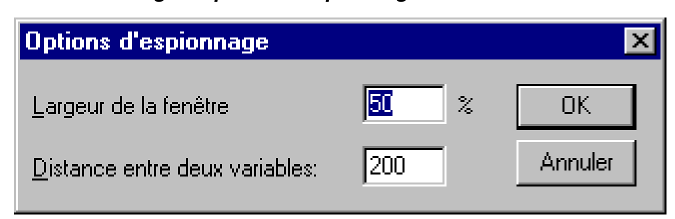
Si vous maintenez le pointeur de souris pendant une courte durée sur une variable, le type et le commentaire de la variable sont affichés dans une info-bulle.

'Extras' 'Options d'espionnage"

Avec cette commande vous pouvez configurer votre fenêtre d'espion. Si l'espionnage est effectué, alors, à l'intérieur des éditeurs littéraux, la fenêtre est scindée en une partie gauche, qui contient le programme, et en une partie droite, dans laquelle sont espionnées toutes les variables contenues dans la ligne de programme correspondante.

Vous pouvez configurer la **Largeur** que la zone d'espionnage doit occuper dans la fenêtre de texte, ainsi que la **Distance** qui doit séparer deux variables d'espionnage dans une même ligne. La spécification de distance 1 correspond à une hauteur de ligne dans la police de caractères choisie.

Boîte de dialogue Options d'espionnage



Positions des points d'arrêt

Il n'est pas possible de définir des points d'arrêt au niveau de chaque ligne, étant donné que plusieurs lignes en langage IL sont synthétisées en interne par **CoDeSys,**de façon à obtenir une seule ligne sous forme de code C. Les points d'arrêt peuvent être définis à tous les endroits du programme où les variables peuvent changer de valeur, ou encore à tous les endroits où le flux du programme subit une déviation (à l'exception des appels de fonction. Dans ce cas il faut définir un point d'arrêt au niveau de la fonction). Il est superflu de définir un point d'arrêt entre les positions précitées, étant donné que les données n'ont pu subir aucune modification depuis le point d'arrêt précédent.

Cela permet les positions de point d'arrêt suivantes, en langage IL:

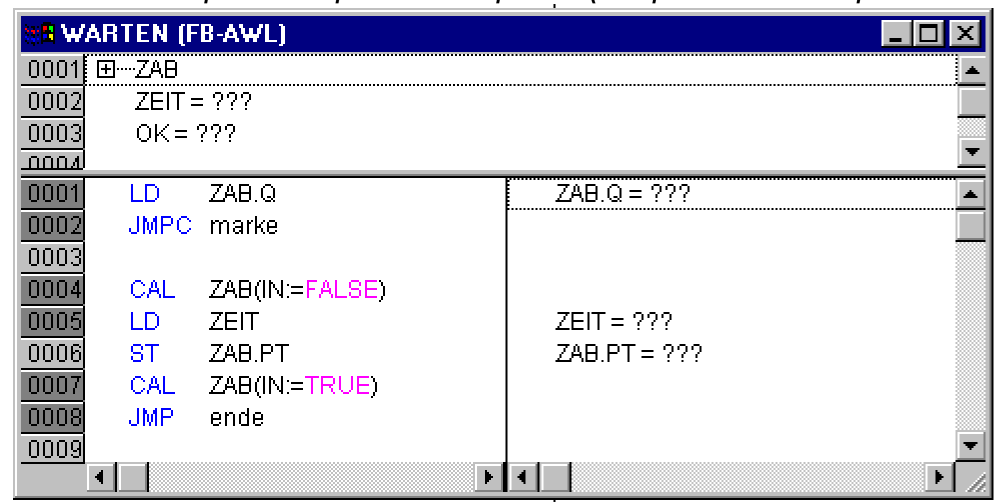
- Au début du module
- €€Au niveau de chaque LD, LDN (ou au niveau d'une étiquette, si un LD est placé directement à la suite de l'étiquette)
- Au niveau de chaque JMP, JMPC, JMPCN
- Au niveau de chaque étiquette
- Au niveau de chaque CAL, CALC, CALCN
- Au niveau de chaque RET, RETC, RETCN
- A la fin du module

Pour le langage ST, les positions de point d'arrêt possibles sont les suivantes:

- Au niveau de chaque affectation
- Au niveau de chaque instruction RETURN et EXIT
- Dans des lignes au sein desquelles des conditions sont évaluées (WHILE, IF, REPEAT)
- A la fin du module

Les positions des points d'arrêt sont caractérisées par une coloration grise plus foncée au niveau du champ de numérotation de ligne.

Editeur IL avec les positions de point d'arrêt disponibles (champs de numérotation plus foncés



Pour définir un point d'arrêt, l'utilisateur doit cliquer avec la souris sur le champ de numérotation de la ligne dans laquelle il souhaite définir un point d'arrêt. Si le champ sélectionné est une position de point d'arrêt, alors la couleur du champ de numérotation de ligne passe du gris foncé au bleu clair et le point d'arrêt est activé dans l'automate programmable.

De façon analogue, pour supprimer un point d'arrêt, il faut cliquer sur le champ de numérotation de la ligne contenant ce point d'arrêt.

Il est également possible d'opter pour la définition ou la suppression de points d'arrêt via le menu ('En Ligne' 'Point d'arrêt actif/inactif'), via la touche de fonction <F9> ou via l'icône appropriée de la barre d'outils.

Comment définit-on un point d'arrêt ?

Pour définir un point d'arrêt, l'utilisateur doit cliquer avec la souris sur le champ de numérotation de la ligne dans laquelle il souhaite définir un point d'arrêt. Si le champ sélectionné est une position de point d'arrêt, alors la couleur du champ de numérotation de ligne passe du gris foncé au bleu clair et le point d'arrêt est activé dans l'automate programmable.

Supprimer un point d'arrêt

De façon analogue, pour supprimer un point d'arrêt, il faut cliquer sur le champ de numérotation de la ligne contenant ce point d'arrêt.

Il est également possible d'opter pour la définition ou la suppression de points d'arrêts via le menu ('En Ligne' 'Point d'arrêt actif/inactif'), via la touche de fonction <F9> ou via l'icône appropriée de la barre d'outils.

Que se produit-il au niveau du point d'arrêt ?

Si l'automate programmable a atteint un point d'arrêt, alors la section contenant la ligne avec le point d'arrêt est visualisée à l'écran. Le champ de numérotation de la ligne qui contient le point d'arrêt est colorée en rouge. Dans l'automate programmable, le traitement du programme utilisateur est suspendu.

Lorsque le programme est arrêté au niveau d'un point d'arrêt, le traitement du programme peut reprendre au moyen de 'En Ligne' 'Démarrer'.

En outre, les commandes 'En Ligne' 'Etape individuelle sur' ou 'En Ligne' 'Etape individuelle dans' ne permettent d'atteindre que la position de point d'arrêt suivante. Si l'instruction sur laquelle on est arrêté est une instruction CAL ou encore si un appel de fonction est présent dans les lignes qui précèdent la position de point d'arrêt suivante, alors, soit on saute cet appel au moyen de **'Etape individuelle sur',** soit on dévie dans le module appelé au moyen de **'Etape individuelle dans'**.

Numéros de ligne de l'éditeur littéral

Les numéros de ligne de l'éditeur littéral indiquent le numéro de chaque ligne de texte présente dans l'implémentation d'un module.

En mode hors ligne, un simple clic sur un numéro de ligne donné suffit pour marquer toute la ligne de texte correspondante.

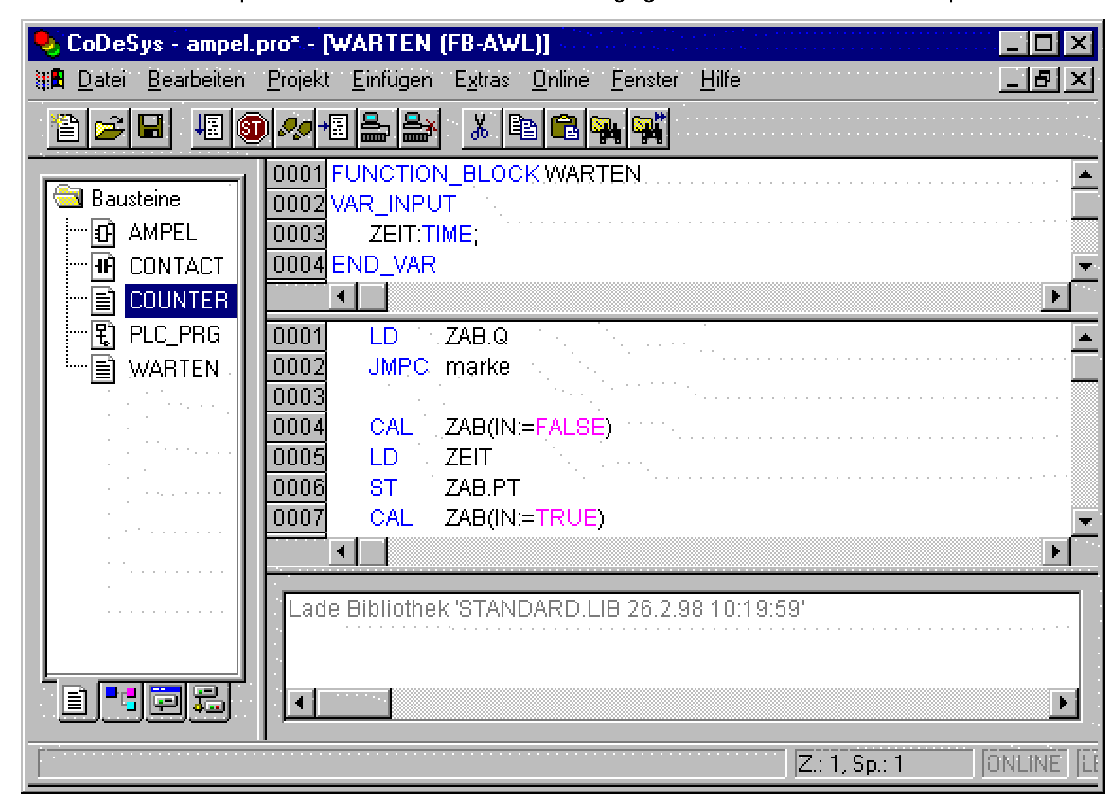
Dans le mode En Ligne, la couleur de fond du numéro de ligne indique l'état de chaque ligne en ce qui concerne les points d'arrêt:

- gris foncé: cette ligne est une position possible de point d'arrêt.
- bleu clair: un point d'arrêt a été défini dans cette ligne.
- rouge: l'exécution du programme a atteint cet endroit.

Dans le mode En Ligne, un simple clic avec la souris modifie l'état de la ligne en ce qui concerne le point d'arrêt.

5.3.2 L'éditeur de la liste d'instructions (IL)

Voici comment se présente un module écrit en langage IL dans l'éditeur correspondant de **CoDeSys**:



Tous les éditeurs pour modules sont constitués d'une partie de déclaration et d'un corps. Ceux-ci sont séparés entre eux par une barre de fractionnement.

L'éditeur IL est un éditeur littéral qui comporte les fonctionnalités courantes des éditeurs littéraux de Windows. Vous trouvez les commandes les plus importantes dans le menu contextuel (touche droite de la souris ou <Ctrl>+<F10>).

Un appel d'instance sur plusieurs lignes est possible. Exemple:

```
CAL CTU_inst(
CU:=%IX10,
PV:=(
LD A
ADD 5
)
)
```

Pour obtenir des informations sur le langage IL, reportez vous au chapitre 2.2.1, "Liste d'instructions (langage IL)".

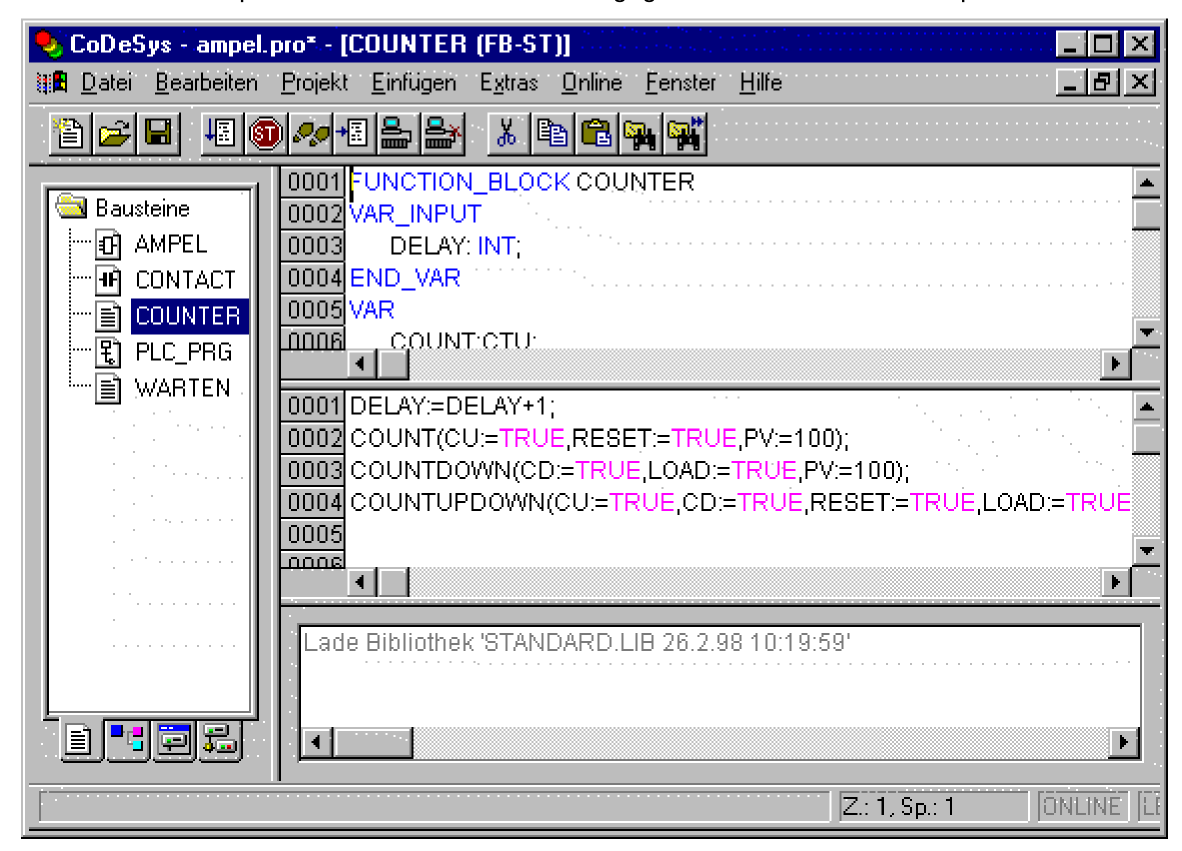
Contrôle de déroulement

Avec la commande 'En Ligne' 'Contrôle de déroulement', vous pouvez insérer, à l'intérieur de l'éditeur IL, un nouveau champ dans la partie gauche de chaque ligne. Ce champ assure la visualisation du contenu de l'accumulateur.

Pour obtenir des informations sur l'éditeur IL en mode En Ligne, reportez vous au paragraphe "Les éditeurs littéraux en mode En Ligne"

5.3.3 L'éditeur du Littéral structuré (ST)

Voici comment se présente un module écrit en langage ST dans l'éditeur correspondant de **CoDeSys**:



Tous les éditeurs pour modules sont constitués d'une partie de déclaration et d'un corps. Ceux-ci sont séparés entre eux par une barre de fractionnement.

L'éditeur ST est un éditeur littéral qui comporte les fonctionnalités courantes des éditeurs littéraux de Windows. Vous trouvez les commandes les plus importantes dans le menu contextuel (touche droite de la souris ou <Ctrl>+<F10>).

Pour obtenir des informations sur l'éditeur ST En Ligne, reportez vous au paragraphe "Les éditeurs littéraux en mode En Ligne"

Pour obtenir des informations sur le langage, reportez vous au chapitre 2.2.1, "Littéral structuré (langage ST)".

5.4 Les éditeurs graphiques

5.4.1 Généralités à propos des éditeurs graphiques

Les éditeurs des langages de programmation graphiques ci après ont de nombreux points communs :diagramme fonctionnel en séquence SFC, langage à contact LD, schémas en blocs fonctionnels en langage FBD et éditeur graphique CFC. Ces points vont être repris dans les paragraphes suivants, et ces différents langages seront amplement détaillés dans les chapitres suivants. Dans les éditeurs graphiques, l'implémentation est assistée par la coloration de la syntaxe.

Zoom

Des objets tels que modules, actions, transitions etc., en langage SFC, LD, FBD, CFC, et apparaissant dans des visualisations, peuvent être agrandis ou réduits au moyen d'une fonction de zoom. Tous les éléments contenus dans la fenêtre de la partie implémentation sont repris, et la partie déclaration reste inchangée.

Chaque objet est affiché par défaut à une valeur de zoom de 100%. La valeur réglée pour le zoom est enregistrée dans le projet comme caractéristique d'objet.

Les impressions de la documentation relative au projet se font toujours en représentation 100%.

Le degré de zoom peut être réglé par le biais d'une liste de sélection dans la barre d'outils. Des valeurs comprises entre 25 et 400% peuvent être directement sélectionnées; des valeurs entre 10 et 500% peuvent être saisies manuellement.

La possibilité de sélection d'une valeur de zoom n'est possible que lorsque le curseur se trouve sur un objet créé à l'aide d'un éditeur graphique ou sur un objet de visualisation.

Les positions du curseur peuvent aussi être sélectionnées à l'état zoomé, et vous pouvez également y déplacer le curseur à l'aide des touches directionnelles. La taille du texte est fonction du degré de zoom et de la taille de la police.

L'exécution de toutes les options disponibles chez les éditeurs (p.ex. l'insertion d'une boîte) en fonction de la position du curseur est possible quel que soit le degré de zoom et en fonction de celui-ci.

Dans le mode En ligne, chaque objet est représenté selon le degré de zoom spécifié, et toutes les fonctions sont disponibles sans restrictions.

Vous pouvez agrandir ou réduire un objet en utilisant IntelliMouse : appuyez sur la touche <Ctrl> et tournez en même temps la roulette vers l'avant ou vers l'arrière.

Réseau

Avec les éditeurs graphiques LD et FBD, le programme est agencé dans une liste de réseaux. Chaque réseau est caractérisé par un numéro courant dans la partie gauche et contient une structure visualisant une expression logique ou arithmétique, un appel de programme, fonction ou bloc fonctionnel, un saut ou une instruction return.

Etiquettes de saut

A chaque réseau correspond une étiquette de saut, qui peut éventuellement être laissée vide. Pour éditer cette étiquette, il faut cliquer sur la première ligne du réseau, située directement à côté du numéro de réseau. Entrez à présent une étiquette, suivie du signe deux-points.

Commentaires de réseau, Mises en Page, 'Extras' 'Options'

Il est possible d'attribuer un commentaire de plusieurs lignes à chaque réseau . La boîte de dialogue 'Options du langage à blocs fonctionnels et du langage à contacts', accessible via la commande 'Extras' 'Options', permet de procéder aux réglages quant aux commentaires. Le champ **Longueur maximale du commentaire** reprend le nombre maximal de lignes disponibles pour un commentaire de réseau (la valeur par défaut est 4). Le champ **Longueur minimale du commentaire** détermine le nombre de lignes qui doit généralement être réservé aux commentaires et à l'affichage de ceux-ci.

Supposons que 2 soit la valeur configurée; deux lignes de commentaires vides sont présentes au début de chaque réseau, après la ligne d'étiquette. Ici, la valeur par défaut est 0, ce qui présente l'avantage que plus de réseaux peuvent être disposés dans la zone d'écran.

Si la taille minimale de commentaire de réseau est supérieure à 0, il suffit simplement, pour entrer un commentaire, de cliquer sur la ligne de commentaire et d'entrer le commentaire. Si ce n'est pas le cas, il faut d'abord sélectionner le réseau qui doit faire l'objet du commentaire, avant d'insérer la ligne de commentaire au moyen de "Insérer" + "Commentaire". Les commentaires sont visualisés en gris, à la différence du texte de programmation.

L'éditeur du langage à contacts vous permet en outre d'attribuer des commentaires pour des contacts ou des bobinages individuels. Activez à cet effet l'option Commentaires par contact et introduisez le nombre souhaité de lignes que vous y prévoyez dans le champ **Lignes pour commentaire de la variable**. Suite à quoi un champ s'affiche, réservé au commentaire relatif au contact ou au bobinage et dans lequel vous pouvez saisir du texte.

Si cette option de commentaire par contact est activée, vous pouvez également définir dans l'éditeur du Langage à contacts le nombre de lignes (**Lignes pour texte de la variable**:) utilisé pour le nom de la variable de contact ou de bobinage, de telle sorte que des noms plus longs puissent le cas échéant être représentés sur plusieurs lignes.

Vous pouvez configurer l'éditeur du langage à contacts de telle manière que des sauts de page soient insérés dans les réseaux dès que la fenêtre ne permet plus de visualiser tous les éléments de ceux-ci. Activez pour ce faire l'option Réseaux avec mise en page.

'Insérer' 'Réseau (derrière)' ou 'Insérer' 'Réseau (devant)'

Raccourci : <Maj>+<T> (Réseau derrière)

Pour insérer un nouveau réseau dans l'éditeur FBD ou dans l'éditeur LD, il faut sélectionner la commande 'Insérer' '**Réseau (derrière)'** ou 'Insérer' '**Réseau (devant)'**, selon que l'on souhaite insérer le nouveau réseau devant ou derrière le réseau actuel. On modifie le réseau actuel en cliquant avec la souris sur le numéro de réseau. Ce réseau est repérable grâce au rectangle en pointillé situé en dessous du numéro. En actionnant la touche <Maj> et en cliquant avec la souris sur un réseau précis, on sélectionne l'ensemble des réseaux situés dans la zone comprise entre le réseau actuel et le réseau sur lequel on a cliqué.

Les éditeurs de réseaux en mode En Ligne

Dans les éditeurs FBD et LD, on peut définir des points d'arrêt uniquement au niveau des réseaux. Le champ de numérotation du réseau pour lequel on a défini un point d'arrêt est visualisé en bleu. Le traitement s'arrête avant le réseau qui contient le point d'arrêt. Dans ce cas le champ de numérotation du réseau est visualisé en rouge. Dans le cas d'une exécution pas à pas, la progression se fait en sautant de réseau en réseau.

Toutes les valeurs sont espionnées au niveau des entrées et sorties des modules (réseaux).

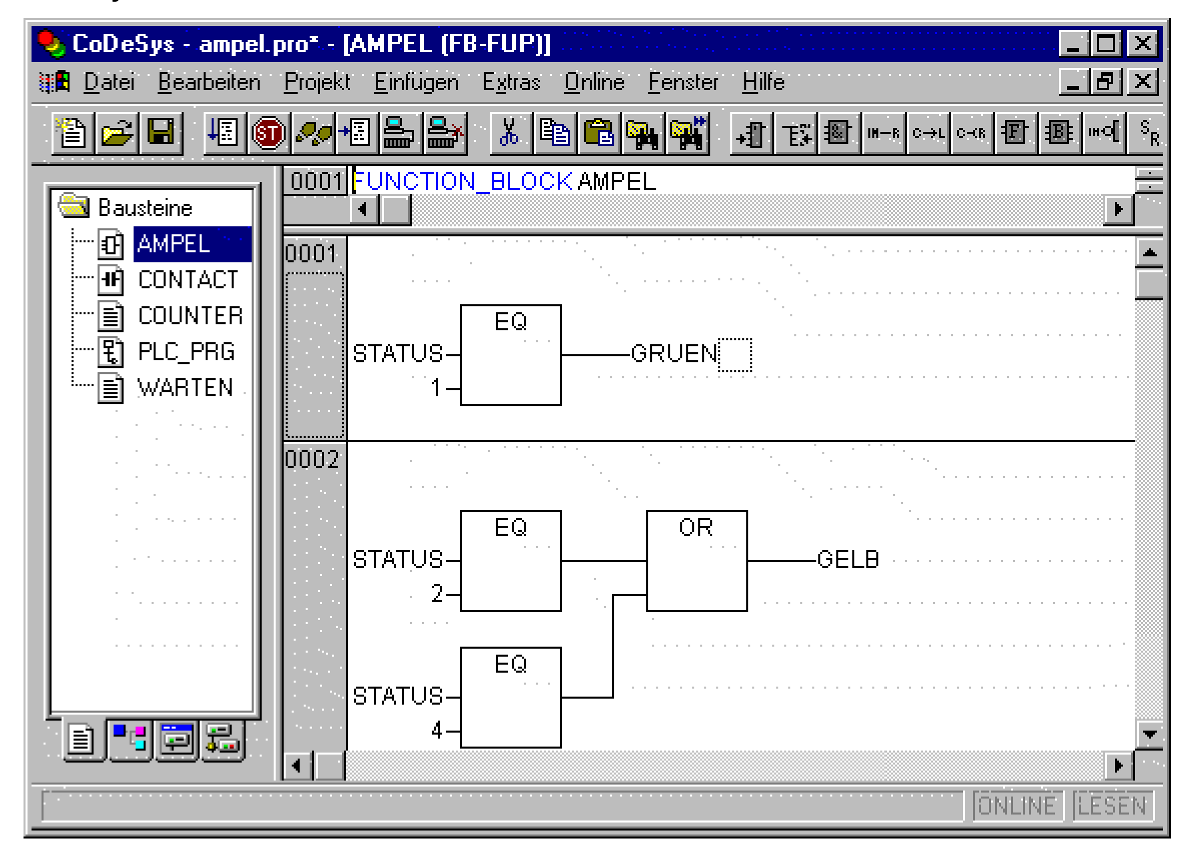
Lors de l'espionnage d'expressions ou de position binaires d'une variable, veuillez noter : Pour les expressions, p.ex. a AND b comme condition de transition ou entrées d'un bloc fonctionnel, la valeur de l'expression complète est toujours affichée (a AND b s'affiche en bleu ou est représentée par ":=TRUE" si a et b ont la valeur TRUE). Dans le cas de positions binaires d'une variable, la position binaire concernée est espionnée (p.ex. a.3 sera affiché en bleu ou représenté avec :=TRUE si a a la valeur 4).

Vous pouvez lancer le contrôle de déroulement au moyen de la commande de menu 'En Ligne' 'Contrôle de déroulement'. Ce contrôle vous permet d'examiner les valeurs actuelles qui sont véhiculées à l'intérieur des réseaux au moyen des éléments de liaison. Lorsque les éléments de liaison ne véhiculent aucune valeur de type booléen, la valeur est affichée dans un champ spécialement inséré à cet effet. Les champs d'espion pour les variables qui ne sont pas utilisées (p.ex. la fonction SEL) sont ombrés en gris. Si les éléments de liaison véhiculent des valeurs de type booléen, et plus précisément la valeur TRUE, alors elles sont colorées en bleu. Cela permet de suivre le flux d'informations lorsque l'automate programmable est en cours d'exécution.

Si vous maintenez le pointeur de souris pendant une courte durée sur une variable, le type et le commentaire de la variable sont affichés dans une info-bulle.

5.4.2 L'éditeur du Schéma en blocs fonctionnels (FBD)

Voici comment se présente un module écrit en langage FBD dans l'éditeur correspondant de **CoDeSys**:



L'éditeur FBD est un éditeur graphique. Il fonctionne avec une liste de réseaux, où chaque réseau contient une structure visualisant une expression logique ou arithmétique, un appel de bloc fonctionnel, un appel d'une fonction, un appel d'un programme, un saut ou une instruction Return.

Notez que pour un module créé En ou Hors ligne dans le langage FBD, il est possible de basculer entre la visualisation au sein de l'éditeur FBD ou LD.

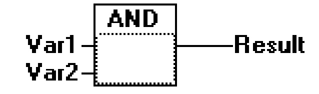
Vous trouvez les commandes les plus importantes dans le menu contextuel (touche droite de la souris ou <Ctrl>+<F10>).

Positions du curseur en langage FBD

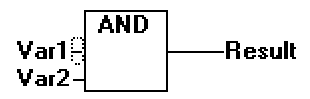
Le curseur peut se positionner sur n'importe quelle élément de texte. Le texte sélectionné est affiché sur fond bleu et peut à présent être modifié.

Par ailleurs, la position actuelle du curseur est caractérisée par un rectangle en pointillés. Voici une énumération des positions de curseurs possibles, illustrées par un Exemple :

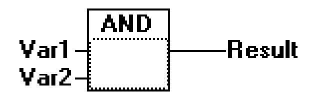
1) Chaque champ de texte (la position du curseur potentielle est encadrée en noir):



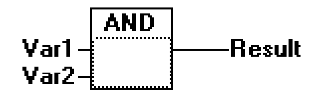
2) Chaque entrée:



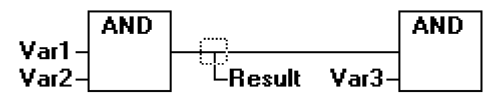
3) Chaque opérateur, fonction ou bloc fonctionnel:



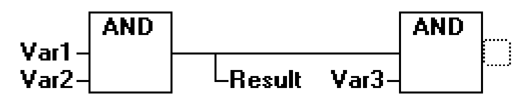
4) Les sorties, si elles sont suivies par une affectation ou par un saut:



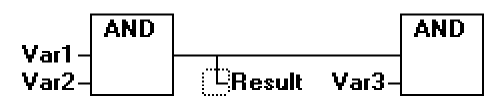
5) La jonction au dessus d'une affectation, d'un saut ou d'une instruction Return:



6) Derrière l'objet situé le plus à droite dans chaque réseau ("dernière position du curseur", il s'agit de la position prise par le curseur lorsqu'un réseau a été sélectionné):



7) La jonction précédant immédiatement une affectation:



Comment positionner le curseur

Le curseur peut être positionné à un endroit précis en cliquant avec la souris ou au moyen du clavier.

Les flèches directionnelles permettent de sauter vers la position du curseur la plus proche, dans la direction choisie. Toutes les positions du curseur peuvent être atteintes de cette façon, y compris les champs de texte. Si le curseur occupe la dernière position, alors la dernière position du réseau précédent ou du réseau suivant peuvent être atteintes par le curseur, respectivement au moyen de la touche directionnelle <haut> et de la touche directionnelle <bas>.

Un réseau vide ne contient que trois points d'interrogation "???". En cliquant derrière ces points d'interrogations, on positionne le curseur sur la dernière position.

'Insérer' 'Module'

Icône : **Raccourci : <Ctrl>+<B>**

Cette commande permet d'insérer des opérateurs, fonctions, blocs fonctionnels et programmes. Tout d'abord, un opérateur "AND" est toujours inséré par défaut. Celui-ci peut être transformé en n'importe quel autre opérateur, fonction, bloc fonctionnel ou programme en sélectionnant et en écrasant le texte type ("AND"). La liste de sélection pour l'édition (<F2>) vous permet de sélectionner le module souhaité. Si le module que vous venez de sélectionner possède un nombre minimal d'entrées différent, alors celles-ci sont rattachées. Si le nouveau module possède un nombre maximal d'entrées moindre, alors les dernières entrées sont supprimées.

Pour les fonctions et les blocs fonctionnels, les noms formels des entrées et sorties sont affichés.

Il existe un champ d'instance éditable au dessus de la boîte pour les blocs fonctionnels. Si, en modifiant le texte type, vous appelez un autre bloc fonctionnel qui n'est pas connu, une boîte d'opérateur s'affiche avec deux entrées ainsi que le type. Si vous sélectionnez le champ d'instance, vous pouvez appeler via la touche <F2> la liste de sélection pour l'édition avec les catégories pour la sélection de variables.

Le nouveau module sera inséré en fonction de la position sélectionnée (voir Position de curseur).

Si une entrée est sélectionnée, alors Boîte est inséré devant cette entrée. La première entrée de cet Box est reliée à la branche située à gauche de l'entrée sélectionnée. La sortie du nouvel Box est reliée à l'entrée sélectionnée.

Si une sortie est sélectionnée, alors Boîte est inséré derrière cette sortie. La première entrée de cet Boîte est reliée à la sortie sélectionnée. La sortie du nouvel Box est reliée à la branche à laquelle était reliée la sortie sélectionnée.

Si un boîte , une fonction ou un bloc fonctionnel est sélectionné (position du curseur 3), alors l'ancien élément est remplacé par le nouvel boîte. Les branches sont reliées autant que possible comme elles l'étaient avant le remplacement. Si l'ancien élément a plus d'entrées que le nouveau, alors les branches qui ne peuvent être reliées à un élément sont supprimées. La même règle s'applique aux sorties.

Si un saut ou une instruction Return est sélectionnée, alors l'opérateur est inséré avant ce saut ou cette instruction Return, selon le cas. La première entrée de boîte est reliée à la branche située à gauche de l'élément sélectionné. La sortie de boîte est reliée à la branche située à droite de l'élément sélectionné.

Si la dernière position de curseur d'un réseau est sélectionnée (position du curseur 6), alors boîte est inséré derrière le dernier élément. La première entrée de boîte est reliée à la branche située à gauche de la position sélectionnée.

Box inséré est obligatoirement un élément AND. Celui-ci peut être transformé en n'importe quel autre boîte en sélectionnant et en écrasant le texte. La liste de sélection pour l'édition vous permet de sélectionner boîte de votre choix dans la liste des boîte supportés. Si le nouvel boîte possède un nombre minimal d'entrées différent, alors les entrées sont rattachées. Si, le nouvel opérateur possède un nombre maximal d'entrées moindre, alors les dernières entrées sont supprimées, ainsi que les branches situées devant celles-ci.

Toutes les entrées de boîte qui n'ont pas pu être reliées sont définies avec le texte "???". Il faut cliquer sur ce texte et le remplacer par la constante ou la variable voulue.

Si une branche est située à droite du module inséré, elle est affectée à la première sortie de module. Pour le reste, les sorties ne font pas l'objet d'une définition.

'Insérer' 'Affectation'

Icône : **Raccourci : <Ctrl>+<A>**

Cette commande insère une affectation.

L'endroit d'insertion est fonction de la position sélectionnée, soit immédiatement avant l'entrée sélectionnée), soit immédiatement après la sortie sélectionnée , soit immédiatement avant la jonction sélectionnée , soit à la fin du réseau .

Lorsqu'une affectation a été insérée, il est ensuite possible de sélectionner le texte existant "???" et de le remplacer par la variable qui doit faire l'objet de l'affectation. Pour ce faire, vous pouvez également utiliser la liste de sélection pour l'édition.

Pour ajouter une nouvelle affectation à une affectation existante, vous devez utiliser la commande 'Insérer' '**Sortie**'.

'Insérer' 'Saut'

Icône : **Raccourci : <Ctrl>+<L>**

Cette commande insère un saut.

L'endroit d'insertion est fonction de la position sélectionnée, soit immédiatement avant l'entrée sélectionnée , soit immédiatement après la sortie sélectionnée , soit immédiatement avant la jonction sélectionnée , soit à la fin du réseau .

Lorsqu'un saut a été inséré, il est ensuite possible de sélectionner le texte existant "???" et de le remplacer par l'étiquette de saut correspondant au saut.

'Insérer' 'Return'

Icône : Raccourci : <Ctrl>+<R>

Cette commande insère une instruction RETURN.

L'endroit d'insertion est fonction de la position sélectionnée, soit immédiatement avant l'entrée sélectionnée , soit immédiatement après la sortie sélectionnée , soit immédiatement avant la jonction sélectionnée, soit à la fin du réseau .

'Insérer' 'Entrée'

Icône : Raccourci : <Ctrl>+<U>

Cette commande insère une entrée d'opérateur. Le nombre des entrées est variable pour un grand nombre d'opérateurs (p.ex. l'élément ADD peut posséder deux ou davantage d'entrées).

Pour ajouter une entrée à un opérateur de ce genre, il faut sélectionner, soit l'entrée devant laquelle une nouvelle entrée doit être insérée (position du curseur 1), soit, lorsqu'une entrée doit être insérée en dessous, l'opérateur lui-même (position du curseur 3).

L'entrée insérée est définie avec le texte "???". Il faut cliquer sur ce texte et le remplacer par la constante ou la variable voulue. Pour ce faire, vous pouvez également utiliser la liste de sélection pour l'édition.

'Insérer' 'Sortie'

Icône :

Cette commande permet d'ajouter à une affectation existante une nouvelle affectation. Cette fonctionnalité permet de créer des échelles d'affectation, c.-à-d. que la valeur actuellement contiguë à la ligne est affectée à différentes variables.

Si la jonction au-dessus d'une affectation ou si la sortie précédant immédiatement celle-ci est sélectionnée, alors une nouvelle affectation est insérée à la suite des affectations existantes.

Si la jonction précédant immédiatement une affectation est sélectionnée (position du curseur 4), alors une nouvelle affectation est insérée devant la première.

La sortie insérée est définie avec le texte "???". Il faut cliquer sur ce texte et le remplacer par la variable de votre choix. Pour ce faire, vous pouvez également utiliser la liste de sélection pour l'édition.

'Insérer' 'Entrée de module'

Raccourci : <Ctrl> + <U>

Cette commande insère une entrée de module. Le nombre des entrées est variable pour un grand nombre d'opérateurs (p.ex. l'élément ADD peut posséder deux ou davantage d'entrées).

Pour ajouter une entrée à un module de ce genre, il faut sélectionner la module lui-même (position du curseur 1).

'Insérer' 'Etiquette'

Icône : Raccourci : <Ctrl>+<L>

Cette commande permet d'insérer une étiquette. Il est possible de sélectionner le texte existant "???" et de le remplacer par une étiquette de saut. En mode En ligne, une étiquette RETURN (RETOUR) est automatiquement insérée pour le marquage de la fin du module.

L'étiquette de saut est insérée à l'aide de la commande 'Insérer' 'Etiquette'.

'Extras' 'Négation'

Icône : **Raccourci : <Ctrl>+<N>**

Cette commande vous permet d'inverser des entrées, des sorties, des sauts ou des instructions RETURN. Le symbole utilisé pour la négation est un petit cercle sur une liaison.

Si une entrée a été sélectionnée , alors la valeur de celle-ci est inversée.

Si une sortie a été sélectionnée , alors la valeur de celle-ci est inversée.

Si un saut ou une instruction Return est marquée, alors la valeur de l'entrée de ce saut ou de cette instruction, selon le cas, est inversée.

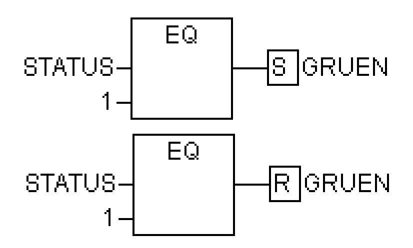
Une négation peut être annulée en inversant la valeur une nouvelle fois.

'Extras' 'Verrouillage/Déverouillage'

Icône : 

Cette commande permet de définir des sorties SET (verrouillage) et des sorties RESET (déverrouillage). Une porte avec une sortie SET est visualisée par un [S], tandis qu'une porte avec une sortie RESET est visualisée par un [R].

Sorties Verrouillage/Déverouillage en langage FBD



Si la porte correspondant à une *sortie SET* fournit comme valeur TRUE, alors la sortie prend la valeur TRUE et elle garde cette valeur même si la porte reprend la valeur FALSE.

Si la porte correspondant à une *sortie RESET* fournit comme valeur TRUE, alors la sortie prend la valeur FALSE et elle garde cette valeur même si la porte reprend la valeur FALSE.

Si la commande est exécutée plusieurs fois, alors la sortie devient tour à tour une sortie SET, une sortie RESET et une sortie normale.

'Extras' 'Affichage'

Cette commande vous permet de sélectionner, pour un module créé avec l'éditeur de schémas en blocs fonctionnels, une représentation par le biais de ce même éditeur ou avec l'éditeur de langage à contacts. Une telle fonction est possible en mode En ligne et en mode Hors ligne.

Zoom sur module appelé

Cette commande est disponible dans le menu contextuel (<F2>) ou dans le menu Extras, pour autant que le curseur se trouve sur le nom du module appelé dans les éditeurs littéraux, ou si la boîte d'un module est marquée dans les éditeurs graphiques. Le zoom ouvre le module concerné dans sa fenêtre d'éditeur. S'il s'agit d'un module issu d'une bibliothèque, alors le gestionnaire de bibliothèques est appelé et le module correspondant est affiché.

Couper, copier, coller et supprimer dans le schéma en blocs fonctionnels

Les commandes '**Couper**', '**Copier**', '**Coller**' et '**Supprimer**' se trouvent dans le menu 'Editer'.

Si une jonction est sélectionnée , alors les affectations, sauts ou instructions RETURN situés en dessous de celle-ci sont coupés, supprimés ou copiés.

Si une boîte est sélectionnée, l'objet sélectionné est coupé, supprimé ou copié, ainsi que toutes les branches contiguës aux entrées, à l'exception de la première (plus élevée) d'entre elles.

Dans les autres cas, la totalité de la branche située devant la position du curseur est coupée, supprimée ou copiée.

Après l'opération de copiage ou de découpage, l'élément supprimé ou copié se trouve dans le presse-papiers et peut être collé autant de fois que l'on veut.

Pour ce faire, il faut d'abord sélectionner la position d'insertion. Les entrées et les sorties sont des positions d'insertion possibles.

Si un boîte a été chargé dans le presse-papiers, (pour mémoire: dans ce cas, toutes les branches contiguës se trouvent dans le presse-papiers, à l'exception de la première), alors la première entrée est reliée à la branche située devant la position d'insertion.

Dans le cas contraire, la totalité de la branche située devant la position d'insertion est remplacée par le contenu du presse-papiers.

Dans tous les cas, le dernier élément collé est relié à la branche située à droite de la position d'insertion.

Remarque : Le découpage et le collage permettent de résoudre le problème suivant. Supposons qu'un opérateur soit collé au milieu d'un réseau. La branche située à droite de l'opérateur est à présent reliée à la première entrée, alors qu'elle devrait être reliée à la deuxième entrée. On sélectionne à présent la première entrée et on exécute la commande 'Editer' 'Couper'. Ensuite, on sélectionne la deuxième entrée et on exécute la commande 'Editer' '**Coller**'. La branche est maintenant rattachée à la deuxième entrée.

Le schéma en blocs fonctionnels en mode En Ligne

Dans le schéma en blocs fonctionnels, il est possible de définir des points d'arrêt uniquement au niveau des réseaux. Si un point d'arrêt a été défini au niveau d'un réseau, alors le champ de numérotation du réseau est visualisé en bleu. Le traitement s'arrête avant le réseau qui contient le point d'arrêt. Dans ce cas, le champ de numérotation du réseau se colore en rouge. Dans le cas d'une exécution pas à pas (étape individuelle), la progression se fait en sautant de réseau en réseau.

La valeur actuelle de chaque variable d'entrée contiguë à un élément de réseau (fonction, programme, instance de bloc fonctionnel ou opérateur) est visualisée.

En double-cliquant sur une variable vous accédez à la boîte de dialogue pour écrire une variable. Exception : si l'entrée d'un bloc fonctionnel est une expression, seule la première variable de l'expression sera espionnée.

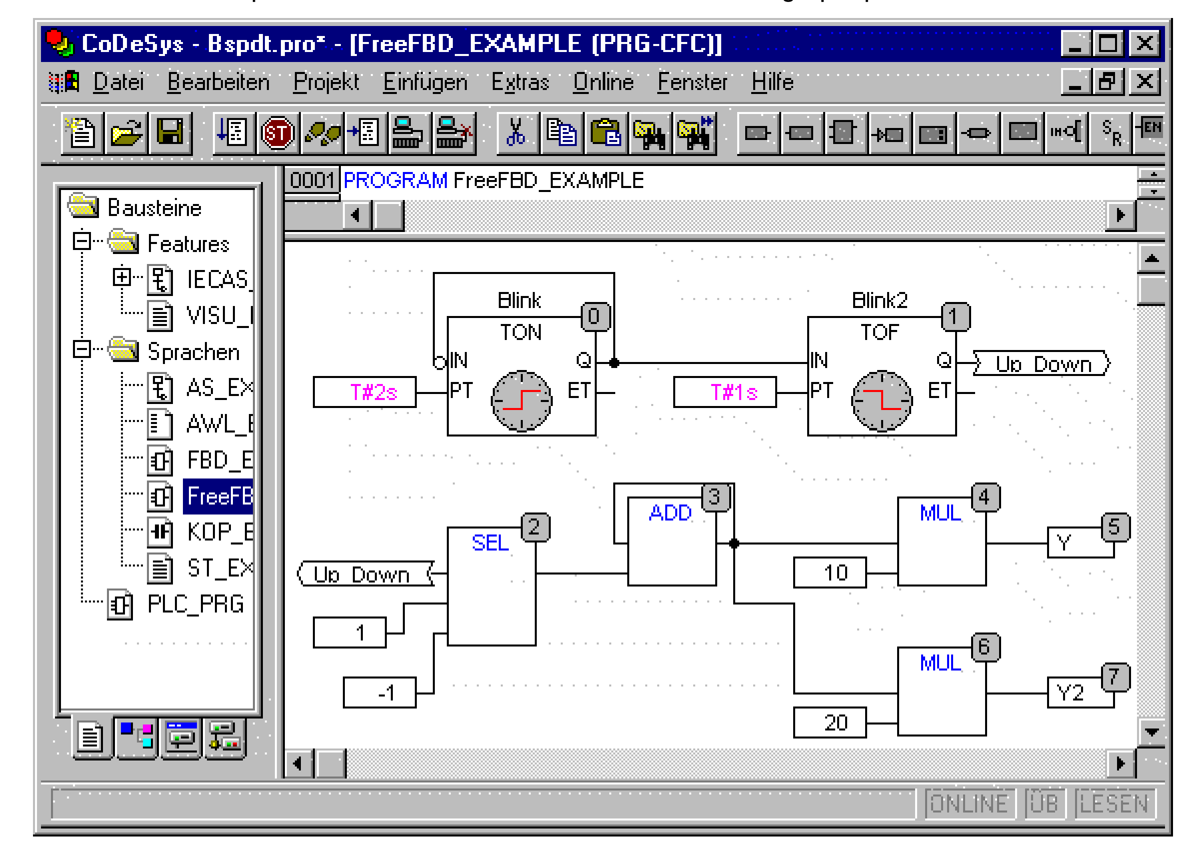
La nouvelle valeur est colorée en rouge et reste inchangée. En donnant la commande 'En Ligne' 'Ecrire valeur des variables', les valeurs sélectionnées sont affectées à toutes les variables et ces dernières se colorent à nouveau en noir.

Vous pouvez lancer le contrôle de déroulement au moyen de la commande de menu 'En Ligne' 'Contrôle de déroulement'. Ce contrôle vous permet d'examiner les valeurs actuelles qui sont véhiculées à l'intérieur des réseaux au moyen des éléments de liaison. Lorsque les éléments de liaison ne véhiculent aucune valeur de type booléen, la valeur est affichée dans un champ spécialement inséré à cet effet. Si les éléments de liaison véhiculent des valeurs de type booléen, et plus précisément la valeur TRUE, alors elles sont colorées en bleu. Cela permet de suivre le flux d'informations lorsque l'automate programmable est en cours d'exécution.

Si vous maintenez le pointeur de souris pendant une courte durée sur une variable, le type et le commentaire de la variable sont affichés dans une info-bulle.

5.4.3 L'éditeur graphique du Schéma en blocs fonctionnels (CFC)

Voici comment se présente un module créé à l'aide de l'éditeur graphique CFC:



L'éditeur graphique du schéma en blocs fonctionnels n'utilisant pas de réseaux, les éléments peuvent être placés librement. La liste d'exécution comprend les éléments suivants: module, entrée, sortie, saut, étiquette, return et commentaire. Les entrées et sorties de ces éléments peuvent être reliées entre elles en déplaçant une liaison au moyen de la souris. L'élément de liaison est dessiné automatiquement. C'est l'élément de liaison le plus court qui est tracé, en tenant compte des liaisons existantes. Lors du déplacement d'éléments, les éléments de liaison sont ajustés automatiquement. Si un élément de liaison ne peut pas être tracé en raison d'un manque de place, alors une ligne rouge est créée entre l'entrée et la sortie correspondante. Dès qu'il existe suffisamment de place, cette ligne est convertie en un élément de liaison.

Avec l'éditeur graphique du schéma en blocs fonctionnels, il est possible d'insérer directement des asservissements, ce qui constitue un avantage par rapport à l'éditeur ordinaire du schéma en blocs fonctionnels.

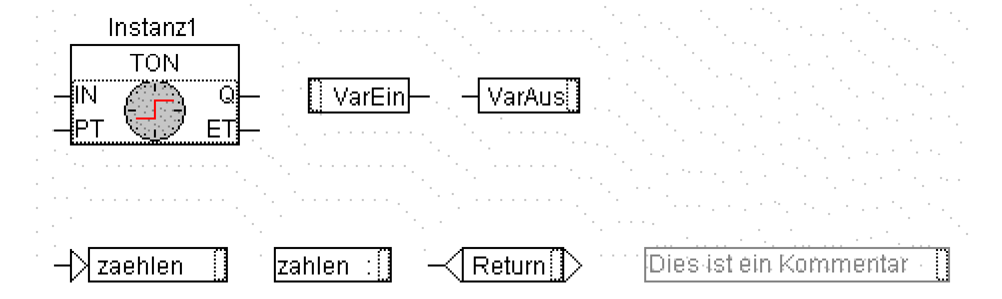
Les commandes principales se trouvent dans le menu contextuel.

Positions du curseur dans l'éditeur graphique CFC

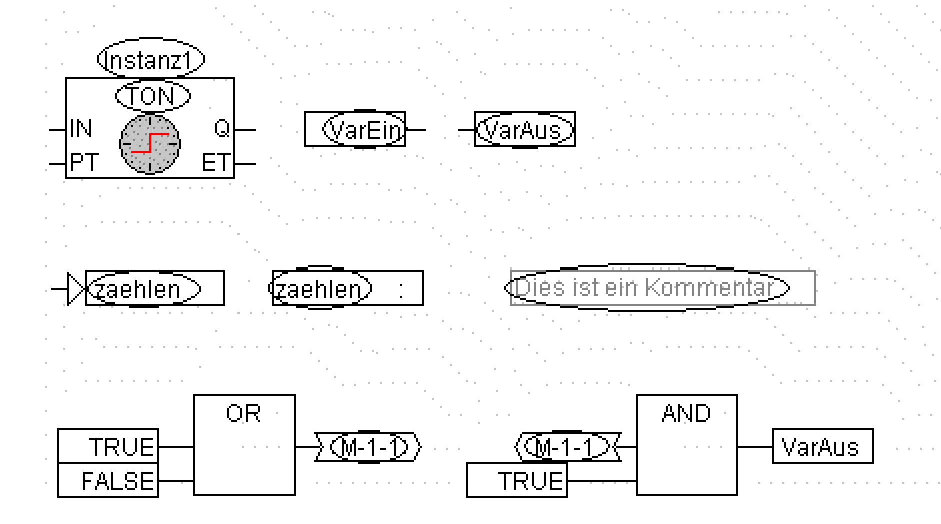
Le curseur peut se positionner sur n'importe quel élément de texte. Le texte sélectionné est affiché sur fond bleu et peut être modifié.

Par ailleurs, la position actuelle du curseur est caractérisée par un rectangle en pointillés. Voici une énumération des positions de curseurs possibles, illustrées par des exemples:

1. Corps des éléments module, entrée, sortie, saut, étiquette, return et commentaire:



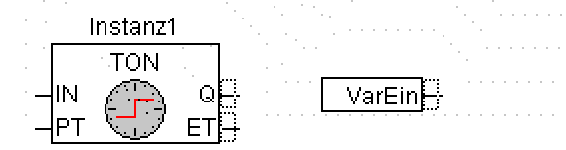
2. Champs de texte des éléments module, entrée, sortie, saut, étiquette et commentaire ainsi que les champs de texte des étiquettes de liaison:



3. Entrée des éléments module, sortie, saut et return:



4. Sortie des éléments module et entrée:



'Insérer' 'Module'



Icône : Raccourci : <Ctrl>+<B>

Cette commande permet d'insérer des opérateurs, fonctions, blocs fonctionnels et programmes. Un opérateur "AND" est toujours inséré par défaut. Celui-ci peut être transformé en n'importe quel autre opérateur, fonction, bloc fonctionnel ou programme en sélectionnant et en écrasant le texte. La liste de sélection pour l'édition vous permet de sélectionner le module de votre choix dans la liste des modules supportés. Si le nouveau module possède un nombre minimal d'entrées différent, alors les entrées sont rattachées. Si le nouveau module possède un nombre maximal d'entrées moindre, alors les dernières entrées sont supprimées.

'Insérer' 'Entrée'

Icône : Raccourci : <Ctrl> + <E>

Cette commande permet d'insérer une entrée. Il est possible de sélectionner le texte existant "???" et de le remplacer par une variable ou une constante. Pour ce faire, vous pouvez également utiliser la liste de sélection pour l'édition.

'Insérer' 'Sortie'

Icône : Raccourci : <Ctrl>+<A>

Cette commande permet d'insérer une sortie. Il est possible de sélectionner le texte existant "???" et de le remplacer par une variable. Pour ce faire, vous pouvez également utiliser la liste de sélection pour l'édition. La valeur existant à l'entrée de la sortie est attribuée à cette variable.

'Insérer' 'Saut'

Icône : Raccourci : <Ctrl>+<J>

Cette commande permet d'insérer un saut. Il est possible de sélectionner le texte existant "???" et de le remplacer par l'étiquette de saut correspondant au saut.

L'étiquette de saut est insérée à l'aide de la commande 'Insérer' 'Etiquette'.

'Insérer' 'Return'

Icône : Raccourci : <Ctrl> + <R>

Cette commande permet d'insérer une instruction RETURN. Veuillez noter qu'une étiquette de saut est automatiquement insérée en mode En ligne, avec la dénomination RETURN, dans la première colonne et après le dernier élément dans l'éditeur ; on passe à cette étiquette lors d'une exécution pas à pas avant de quitter le module.

'Insérer' 'Commentaire'

Icône : Raccourci : <Ctrl> + <K>

Cette commande permet d'insérer un commentaire.

Vous insérez une nouvelle ligne dans le commentaire en actionnant <Ctrl> + <Entrée>.

'Insérer' 'Entrée de module'

Raccourci : <Ctrl> + <U>

Cette commande insère une entrée de module. Le nombre des entrées est variable pour un grand nombre d'opérateurs (p.ex. l'élément ADD peut posséder deux ou davantage d'entrées).

Pour ajouter une entrée à un module de ce genre, il faut sélectionner la module lui-même (position du curseur 1).

Insérer' 'Insérer une entrée CFC, 'Insérer une sortie CFC'

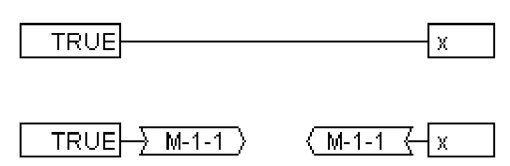
Icône :

Ces commandes sont disponibles dès qu'une macro est ouverte pour édition. Elles servent à l'insertion de pins d'entrée ou de sortie servant d'entrée et de sortie pour la macro. Elles se différencient des entrées ou sorties normales de modules de par leur représentation et de par le fait qu'elles n'ont pas d'index de position.

'Extras' 'Connecteur'

Des liaisons peuvent aussi être visualisées à l'aide de connecteurs (étiquettes de liaisons) plutôt qu'à l'aide d'éléments de liaison. Dans ce cas, la sortie et l'entrée correspondante sont dotées d'un connecteur qui est désigné par un nom univoque.

S'il existe déjà une liaison entre deux éléments et que l'on souhaite visualiser à présent cette liaison à l'aide de connecteurs, il faut commencer par marquer la sortie de l'élément de liaison (position du curseur 3), puis sélectionner l'élément de menu 'Extras' '**Etiquette de liaison'**. La figure suivante représente une liaison avant et après sélection de cet élément de menu.



Le programme attribue par défaut un nom univoque à chaque connecteur, commençant par la lettre M. Ce nom peut néanmoins être modifié. Le nom de connecteur est enregistré comme paramètre de sortie, mais peut cependant être édité aussi bien comme entrée que comme sortie.

Nous attirons votre attention sur le fait que le nom de connecteur est considéré comme étant une propriété d'une sortie d'une liaison et qu'il est enregistré avec cette sortie.

1. Editer le nom de connecteur au niveau de la sortie:

Si le texte à l'intérieur du connecteur est remplacé, le nouveau nom de connecteur est pris en compte par tous les connecteurs correspondants au niveau des entrées. Cependant, il n'est pas possible de choisir un nom qui est déjà attribué à **une** autre étiquette de liaison. Cela permet de conserver des noms de connecteur univoques. Dans ce cas, un message adéquat s'affiche à l'écran.

2. Editer le nom de connecteur au niveau de l'entrée:

Si le texte à l'intérieur du connecteur est remplacé, il le sera également dans l'étiquette de liaison correspondante dans l'autre module.

Pour rétablir une liaison ordinaire à partir d'une liaison visualisée à l'aide de connecteurs, marquez les sorties des liaisons (position du curseur 4) et sélectionnez à nouveau l'option de menu 'Extras' '**Etiquettes de liaison**'.

'Extras' 'Inversion'

Icône : Raccourci : <Ctrl> + <N>

Cette commande vous permet d'inverser des entrées (négation), des sorties, des sauts ou des instructions RETURN. Le symbole utilisé pour la négation est un petit cercle sur une liaison.

Si une entrée d'un module, d'une sortie, d'un saut ou d'un return a été sélectionnée (position du curseur 3, alors la valeur de celle-ci est inversée.

Si une sortie d'un module ou d'une entrée a été sélectionnée (position du curseur 4), alors la valeur de celle-ci est inversée.

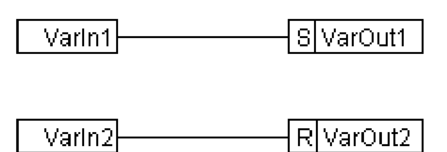
Une négation peut être annulée en inversant la valeur une nouvelle fois.

'Extras' 'Set/Reset'

Icône : Raccourci : <Ctrl> + <T>

Cette commande n'est exécutable que pour les entrées sélectionnées des éléments de type sortie (position du curseur 3).

Le symbole S représente SET et le symbole R représente RESET.



VarOut1 devient TRUE lorsque VarIn1 fournit TRUE. VarOut1 garde cette valeur même si VarIn1 reprend la valeur FALSE.

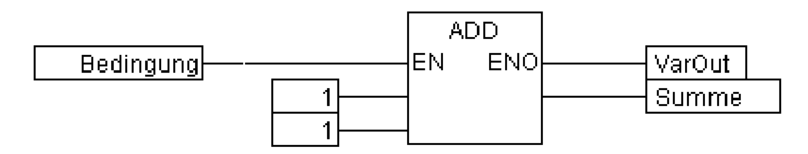
VarOut2 devient FALSE lorsque VarIn2 fournit TRUE. VarOut2 garde cette valeur même si VarIn2 reprend la valeur **FALSE**.

Si la commande est exécutée plusieurs fois, alors la sortie prend tour à tour l'état SET, l'état RESET et l'état normal.

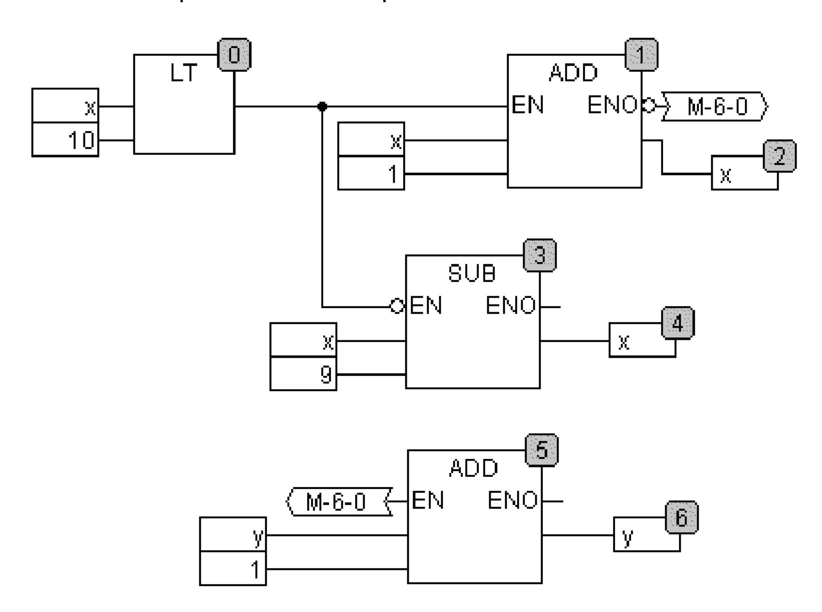
'Extras' 'EN/ENO'

Icône : **Raccourci : <Ctrl> + <0>**

Cette commande permet d'ajouter une entrée d'activation EN (Enable In) supplémentaire et une sortie booléenne ENO (Enable Out) au module sélectionné.



Dans cet exemple, la fonction ADD sera uniquement exécutée si la variable booléenne "Bedingung" (condition) à la valeur TRUE. VarOut prendra également la valeur TRUE après l'exécution de la fonction ADD. Si la variable "Bedingung" (condition) est égale à FALSE, alors la fonction ADD n'est pas exécutée et la valeur FALSE est affectée à VarOut. L'exemple ci-dessous montre comment la valeur d'ENO peut être utilisée pour d'autres modules.



Pour cela, x doit être initialisé avec la valeur 1 et y avec la valeur 0. Les numéros qui figurent dans le coin droit des modules indiquent l'ordre d'exécution.

x est incrémenté jusqu'à ce qu'il prenne la valeur 10. Dès lors, la sortie du module LT(0) fournit comme valeur FALSE, ce qui entraîne l'exécution de SUB(3) et de ADD(5). Par conséquent, x prend la valeur 1 tandis que y est incrémenté d'une unité. Ensuite, c'est le module LT(0) qui est à nouveau exécuté aussi longtemps que x est inférieur à 10. Par conséquent, y comptabilise le nombre de fois que x parcourt les valeurs de 1 à 10.

'Extras' 'Caractéristiques...'

Dans l'éditeur graphique du schéma en blocs fonctionnels, les paramètres d'entrée constants (VAR_INPUT CONSTANT) des fonctions et des blocs fonctionnels ne sont pas affichés directement. Ces paramètres peuvent être affichés et leur valeur peut être modifiée en sélectionnant le corps du module concerné (Position du curseur 1 et la commande 'Extras' '**Propriétés…**', ou simplement en double-cliquant sur le corps. Pour cela, la boîte de dialogue suivante s'affiche:

Boîte de dialogue Propriétés

Traiter les paramètres

Nom	Valeur
a	1
b	2
c	3

OK

Annuler

Les valeurs des paramètres d'entrée constants (VAR_INPUT CONSTANT) peuvent être modifiées. Pour cela, il faut sélectionner la valeur du paramètre dans la colonne Valeur. Cette valeur peut être éditée en cliquant une nouvelle fois avec la souris ou en appuyant sur la <Barre d'espacement>. Pour confirmer la modification d'une valeur, il faut appuyer sur la touche <Entrée>; pour rejeter les modifications, appuyez sur la touche <Echap>. Pour enregistrer toutes les modifications, appuyez sur le bouton **OK**.

Sélectionner des éléments

Pour sélectionner un élément, cliquez avec la souris sur le corps de l'élément (position de curseur 1).

Pour sélectionner plusieurs éléments, soit vous maintenez la touche <Maj> enfoncée et vous cliquez successivement avec la souris sur chacun des éléments correspondants, soit vous créez une fenêtre recouvrant les éléments à sélectionner, en déplaçant la souris et en maintenant la touche gauche enfoncée.

La commande 'Extras' **'Marquer tout'** permet de sélectionner tous les éléments.

Déplacer des éléments

Vous pouvez déplacer un ou plusieurs éléments sélectionnés en maintenant la touche <Maj> enfoncée et en actionnant les touches directionnelles. Vous pouvez également déplacer des éléments au moyen de la souris, en maintenant la touche gauche de la souris enfoncée. Pour déposer ces éléments, relâchez la touche gauche de la souris, à condition que les éléments ne recouvrent pas d'autres éléments et qu'ils n'excèdent pas les dimensions prévues pour l'éditeur. Dans ce cas, les éléments marqués retournent à leur emplacement de départ et un signal d'avertissement retentit.

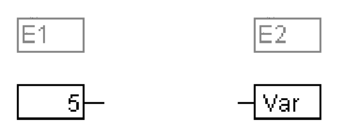
Copier des éléments

Un ou plusieurs éléments sélectionnés peuvent être copiés au moyen de la commande 'Editer' 'Copier' ou insérés avec la commande 'Editer' 'Coller'.

Etablir des liaisons

Une entrée d'élément peut être reliée à une et à une seule sortie d'élément. Une sortie d'élément peut être reliée à plusieurs entrées d'éléments.

Il existe plusieurs façons de relier une entrée d'un élément E2 à une sortie d'un élément E1.



Cliquez avec la touche gauche de la souris sur la sortie de l'élément E1 (position du curseur 4), maintenez la touche gauche de la souris enfoncée, déplacez le pointeur de la souris jusqu'à l'entrée de l'élément E2 (position du curseur 3) et relâchez la touche gauche de la souris à cet endroit. Pendant le déplacement de la souris, une liaison est dessinée entre la sortie de l'élément E1 et le pointeur de la souris.

Cliquez avec la touche gauche de la souris sur l'entrée de l'élément E2, maintenez la touche gauche de la souris enfoncée, déplacez le pointeur de la souris jusqu'à la sortie de l'élément E1 et relâchez la touche gauche de la souris à cet endroit.

Déplacez soit l'élément E1, soit l'élément E2 (position du curseur 1) et déposez-le en relâchant la touche gauche de la souris de telle façon que la sortie de l'élément E2 et l'entrée de l'élément E1 soient en contact.

Dans le cas où l'élément E2 est un module avec une entrée libre, il est possible de créer une liaison en effectuant un déplacement à partir d'une des sorties de E1 jusque dans le corps de l'élément E2. Dès que vous relâchez la touche de la souris, une liaison avec l'entrée libre de E2 la plus haute est créée. Si le module E2 ne possède aucune entrée libre, mais se trouve être un opérateur auquel il est possible d'ajouter une entrée, alors une nouvelle entrée est créée automatiquement.

En appliquant les mêmes méthodes, il est possible de relier entre elles la sortie et l'entrée d'un module (asservissement). Pour établir une liaison entre deux pins, il suffit de cliquer à l'aide du bouton gauche de la souris sur l'un d'entre eux, de se déplacer tout en maintenant le bouton enfoncé jusqu'au pin souhaité, où vous pouvez relâcher le bouton. Si vous quittez la zone de travail de l'éditeur durant cette manœuvre, vous faites automatiquement défiler l'écran. Un contrôle du type se produit lors de la liaison dans le cas de types de données simples. Si les types des pins que vous souhaitez relier sont incompatibles entre eux, le curseur se modifie et affiche « Interdit ». Avec des types de données complexes, il n'y a pas de contrôle.

Supprimer des liaisons

Il existe plusieurs façons de supprimer une liaison entre la sortie d'un élément E1 et l'entrée d'un élément E2.

Marquez la sortie de E1 (position du curseur 4) et appuyez sur la touche <Suppr> ou exécutez la commande 'Editer' 'Supprimer'. Si la sortie de E1 est reliée à plusieurs entrées, alors les liaisons correspondantes sont supprimées.

Marquez l'entrée E2 (position du curseur 4) et appuyez sur la touche <Suppr> ou exécutez la commande 'Editer' '**Supprimer**'.

Marquez avec la souris l'entrée de E2 et, tout en maintenant la touche gauche de la souris enfoncée, écartez la liaison de l'entrée de E2. Pour supprimer la liaison, il suffit de relâcher la touche gauche de la souris dans une zone libre.

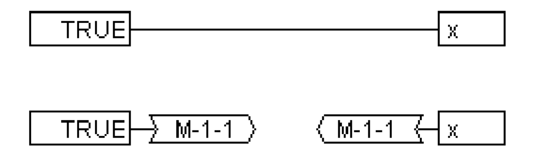
Modifier des liaisons

Il est possible de convertir facilement une liaison entre la sortie d'un élément E1 et l'entrée d'un élément E2 en une liaison entre la sortie de E1 et une entrée d'un élément E3. Pour ce faire, cliquez avec la souris sur l'entrée de E2 (position du curseur 3) et, tout en maintenant la touche gauche de la souris enfoncée, déplacez le pointeur de la souris jusqu'à l'entrée de E3 et relâchez la touche de la souris à cet endroit.

'Extras' 'Connecteur'

Des liaisons peuvent aussi être visualisées à l'aide de connecteurs (étiquettes de liaisons) plutôt qu'à l'aide d'éléments de liaison. Dans ce cas, la sortie et l'entrée correspondante sont dotées d'un connecteur qui est désigné par un nom univoque.

S'il existe déjà une liaison entre deux éléments et que l'on souhaite visualiser à présent cette liaison à l'aide de connecteurs, il faut commencer par marquer la sortie de l'élément de liaison (position du curseur 3), puis sélectionner l'élément de menu 'Extras' '**Etiquette de liaison'**. La figure suivante représente une liaison avant et après sélection de cet élément de menu.



Le programme attribue par défaut un nom univoque à chaque connecteur, commençant par la lettre M. Ce nom peut néanmoins être modifié. Le nom de connecteur est enregistré comme paramètre de sortie, mais peut cependant être édité aussi bien comme entrée que comme sortie.

Nous attirons votre attention sur le fait que le nom de connecteur est considéré comme étant une propriété d'une sortie d'une liaison et qu'il est enregistré avec cette sortie.

1. Editer le nom de connecteur au niveau de la sortie: M-1-1

Si le texte à l'intérieur du connecteur est remplacé, le nouveau nom de connecteur est pris en compte par tous les connecteurs correspondants au niveau des entrées. Cependant, il n'est pas possible de choisir un nom qui est déjà attribué à **une** autre étiquette de liaison. Cela permet de conserver des noms de connecteur univoques. Dans ce cas, un message adéquat s'affiche à l'écran.

2. Editer le nom de connecteur au niveau de l'entrée: M-1-1

Si le texte à l'intérieur du connecteur est remplacé, il le sera également dans l'étiquette de liaison correspondante dans l'autre module.

Pour rétablir une liaison ordinaire à partir d'une liaison visualisée à l'aide de connecteurs, marquez les sorties des liaisons (position du curseur 4) et sélectionnez à nouveau l'option de menu 'Extras' '**Etiquettes de liaison**'.

Insérer des entrées / des sorties "à la volée"

Si vous sélectionnez un seul pin d'entrée ou de sortie d'un élément, vous pouvez insérer directement au clavier l'élément d'entrée ou de sortie correspondant par le biais d'une chaîne de caractères et remplir son champ d'édition à l'aide de cette chaîne de caractères.

Ordre d'exécution

Dans l'éditeur graphique du schéma en blocs fonctionnels, un numéro d'exécution est attribué à chaque élément de type module, sortie, saut, return et étiquette. Cela définit l'ordre dans lequel les différents éléments sont pris en compte pendant l'exécution du programme.

Le numéro est attribué automatiquement selon un ordre topologique, lors de l'insertion de l'élément (de gauche à droite et de haut en bas). Si l'ordre a déjà été modifié, le nouvel élément reçoit le numéro de l'élément suivant dans l'ordre topologique et tous les numéros plus élevés sont incrémentés d'une unité.

Le numéro d'un élément est conservé lors du déplacement de l'élément.

L'ordre influe sur le résultat et doit être modifié dans des cas précis.

Si l'ordre est affiché, alors le numéro d'exécution apparaît dans le coin supérieur droit de l'élément auquel il se rapporte.

'Extras' 'Ordre' 'Affichage'

Cette commande permet d'activer ou de désactiver l'affichage de l'ordre d'exécution. L'ordre d'exécution est affiché par défaut (dans ce cas, un crochet est mis devant l'élément de menu).

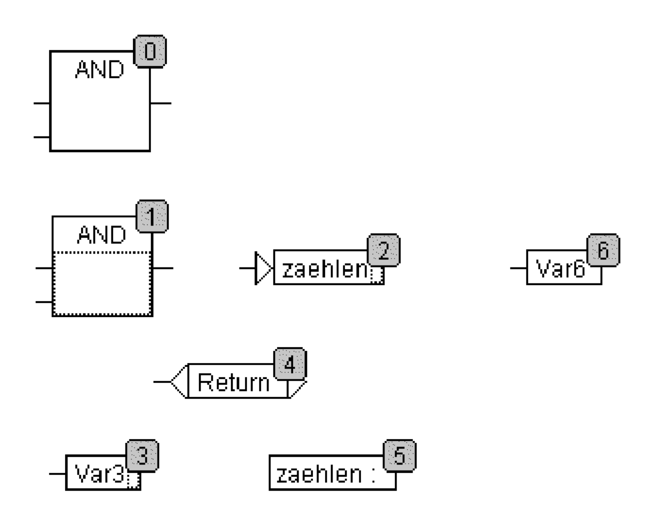
Pour les éléments de type module, sortie, saut, return et étiquette, le numéro d'exécution apparaît dans le coin supérieur droit de l'élément auquel il se rapporte.

'Extras' 'Ordre' 'Ordonner selon l'ordre topologique'

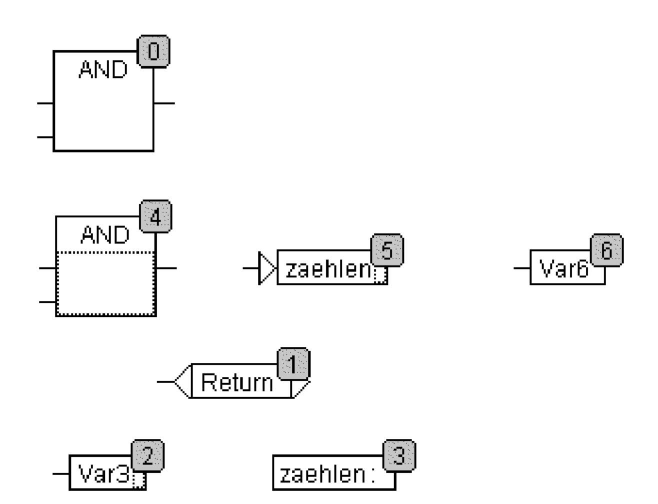
Les éléments sont ordonnés selon l'ordre topologique si l'exécution a lieu de gauche à droite et de haut en bas. Autrement dit, pour des éléments en ordre topologique, le numéro augmente de la

gauche vers la droite et du haut vers le bas. Les liaisons ne jouent aucun rôle. Seule la disposition des éléments compte.

Si la commande 'Extras' '**Ordre**' '**Ordonner selon l'ordre topologique**' est exécutée, alors tous les éléments **sélectionnés** sont ordonnés selon l'ordre topologique. Dans le même temps, tous les éléments de la sélection sont extraits de la liste d'exécution. Ensuite, les éléments de la sélection sont réinsérés séparément dans la liste d'exécution restante, en commençant en bas à droite et en terminant en haut à gauche. Chaque élément sélectionné est inséré devant l'élément suivant dans l'ordre topologique, c'est-à-dire devant l'élément qui serait exécuté ensuite si tous éléments étaient ordonnés selon l'ordre topologique. Illustrons cela à l'aide d'un exemple.



Les éléments portant les numéros 1, 2 et 3 sont sélectionnés. Si la commande '**Ordonner selon l'ordre topologique**' est sélectionnée, alors les trois éléments sélectionnés sont extraits de la liste d'exécution. Ensuite, Var 3, le saut et l'opérateur AND sont réinsérés, dans cet ordre. Var3 est placé devant l'étiquette et reçoit le numéro 2. Ensuite, le saut est inséré et reçoit provisoirement le numéro 4 qui se transforme en 5 après insertion de l'opérateur AND. On obtient un nouvel ordre d'exécution qui se présente comme ceci:



Lors de la disposition d'un nouveau module, celui-ci est placé par défaut devant l'élément suivant selon l'ordre topologique, au sein de la liste d'exécution.

'Extras' 'Ordre' 'Avancer le séquencement par un'

Cette commande permet de faire avancer tous les éléments sélectionnés d'une place dans l'ordre d'exécution, à l'exception de l'élément qui est situé en tête de l'ordre d'exécution.

'Extras' 'Ordre' 'Remettre le séquencement par un'

Cette commande permet de faire reculer tous les éléments sélectionnés d'une place dans l'ordre d'exécution, à l'exception de l'élément qui est situé à la fin de l'ordre d'exécution.

'Extras' 'Ordre' 'Au début'

Cette commande permet de déplacer tous les éléments sélectionnés au début de l'ordre d'exécution, tout en conservant l'ordre d'exécution de ces éléments. L'ordre d'exécution des éléments non sélectionnés est également conservé.

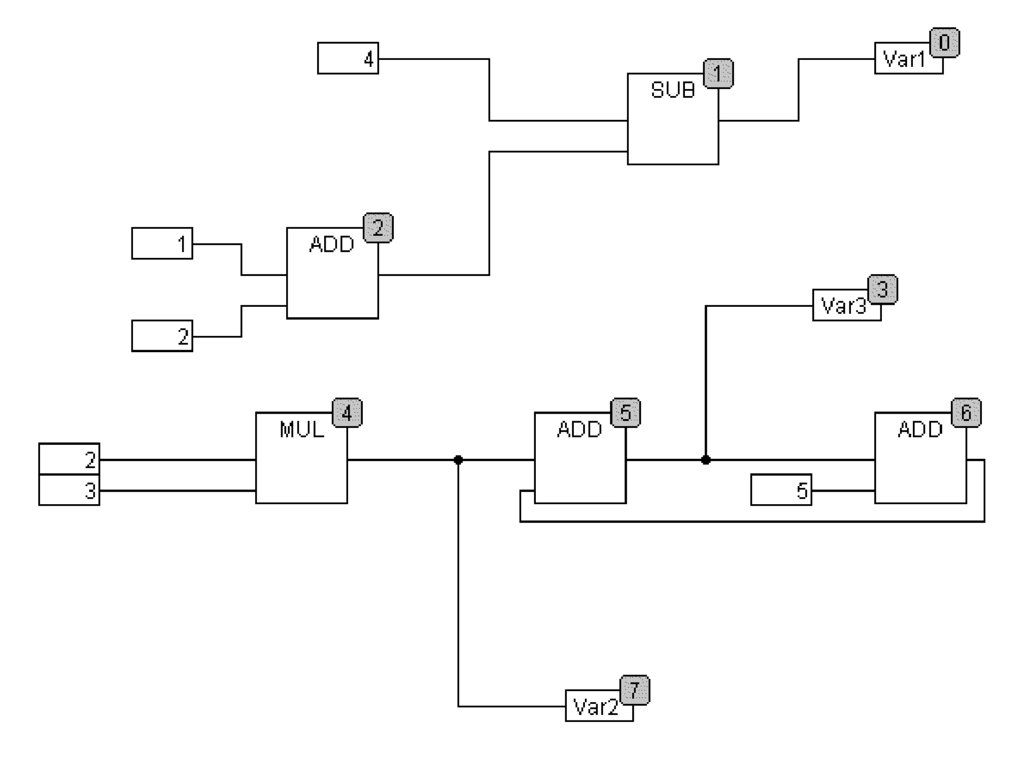
'Extras' 'Ordre' 'A la fin'

Cette commande permet de déplacer tous les éléments sélectionnés à la fin de l'ordre d'exécution, tout en conservant l'ordre d'exécution de ces éléments. L'ordre d'exécution des éléments non sélectionnés est également conservé.

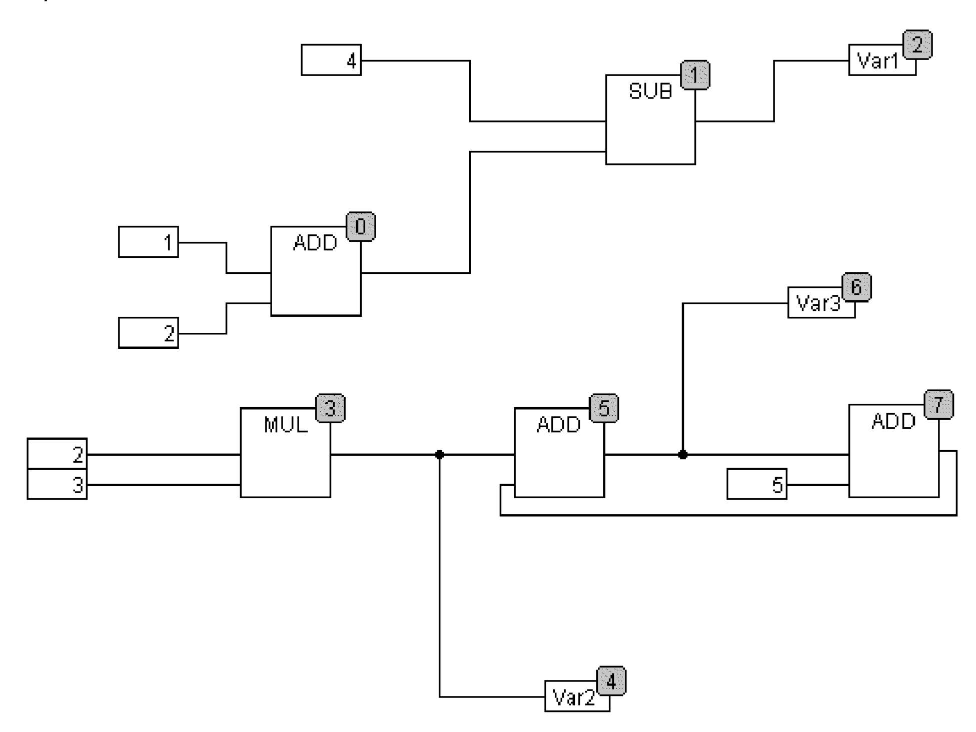
'Extras' 'Ordre' 'Ordonner selon flux des données'

Cette commande est appliquée **à tous** les éléments. L'ordre d'exécution est déterminé par le flux des données des éléments et non par leur disposition.

La figure ci-dessous montre des éléments ordonnés selon l'ordre topologique:



Après sélection de cette commande, les éléments sont ordonnés comme suit:



Lorsque cette commande est sélectionnée, tous les éléments sont triés dans une première étape selon l'ordre topologique. Ensuite, une nouvelle liste d'exécution est constituée. En s'appuyant sur les valeurs des entrées connues, on détermine quel est le prochain élément non numéroté susceptible d'être exécuté. Dans le 'réseau' supérieur, le module ADD, par exemple, est susceptible d'être exécuté immédiatement, car les valeurs à ses entrées sont connues (1 et 2). Le module SUB ne pourra être exécuté qu'ensuite, étant donné que le résultat de ADD doit être connu, etc.

Les asservissements sont cependant insérés en dernier.

Lorsque l'ordre est établi en fonction du flux des données, cela présente l'avantage qu'une box de sortie reliée à la sortie d'un module est toujours placée juste après celui-ci, ce qui n'est pas toujours le cas avec l'ordre topologique. Par conséquent, il se peut que l'ordre topologique ne fournisse pas toujours le même résultat que l'ordre dépendant du flux des données. Cette dernière hypothèse se vérifie dans l'exemple ci-dessus.

'Extras' 'Créer Macro'

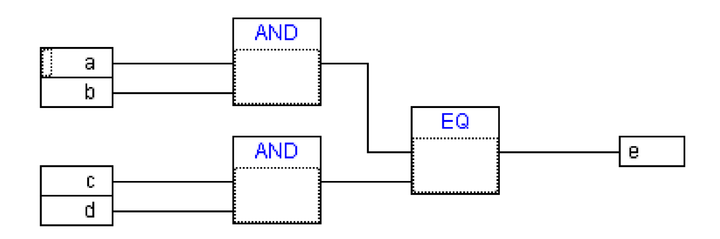
Icône :

À l'aide de cette commande, vous pouvez regrouper plusieurs modules sélectionnés simultanément en un bloc auquel vous pouvez donner un nom en tant que macro. Vous ne pouvez reproduire des macros qu'au moyen de la fonction copier-coller, chaque copie représentant une macro dont vous pouvez choisir le nom en toute liberté. De ce fait, les macros ne sont pas des références. Toutes les liaisons qui sont « décapitées » par la création de la macro créent des pins d'entrée ou de sortie à la macro. Les liaisons vers des entrées créent des pins d'entrée. Un nom par défaut apparaît à côté du pin sous la forme In<n>. Dans le cas de liaisons vers des sorties, le nom est Out<n>. Les liaisons concernées qui disposaient d'étiquettes de liaison avant la création de la macro gardent cette étiquette de liaison sur le PIN de la macro.

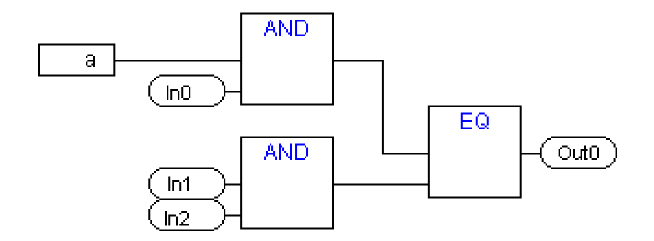
Une macro reçoit tout d'abord le nom par défaut „MACRO". Celui-ci peut être modifié dans le champ du nom de la macro. Si vous éditez la macro, son nom sera affiché dans la barre de titre de la fenêtre d'éditeur, ajouté au nom du module.

Exemple :

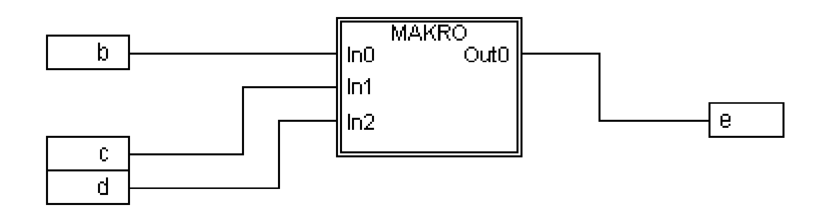
Sélection



Macro :



Dans l'éditeur :



'Extras' 'Sauter dans le macro'

Icône :

Par le biais de cette commande ou en double-cliquant sur le corps de la macro, cette même macro est ouverte dans la fenêtre d'éditeur du module correspondant en vue d'une édition. Le nom de la macro est rattaché au nom du module dans la barre de titre.

Vous pouvez utiliser les boîtes de pins obtenues lors de la création de manière similaire aux entrées et sorties du module. Elles peuvent donc être glissées, effacées, ajoutées, etc. Elles ne se différencient qu'au niveau de la visualisation et ne disposent pas d'un index de position. Pour l'ajout, vous pouvez utiliser les boutons (entrée) ou (sortie) qui vous sont proposées dans la barre d'outils. Les boîtes de pins ont des coins arrondis. Le texte de la boîte de pin correspond au nom du pin dans la visualisation de la macro.

L'ordre des pins sur la boîte de la macro est établi en fonction de l'ordre selon lequel les éléments de la macro sont exécutés. Un numéro d'ordre moins élevé avant un plus élevé, un pin supérieur avant un inférieur.

L'ordre d'exécution au sein de la macro est fermé, c.-à-d. la macro est considérée comme un bloc et ce au niveau de l'emplacement de la macro dans le module principal. Les commandes quant à la manipulation de l'ordre d'exécution n'ont d'effet qu'au sein de la macro.

'Extras' 'Expansion de la macro'

Cette commande permet d'étendre la macro sélectionnée et d'insérer à la position de la macro dans le module les éléments qu'elle contient. Les liaisons vers les pins de la macro sont à nouveau considérées comme des liaisons aux entrées ou sorties des éléments. Si vous ne pouvez étendre la macro en raison d'un manque de place à l'emplacement de la boîte de macro, cette macro sera déplacée vers la droite ou vers le bas jusqu'à ce qu'il y ait suffisamment de place.

Remarque : Si le projet est sauvegardé sous la version de projet 2.1, toutes les macros sont automatiquement étendues. Avant la conversion en d'autres langages, toutes les macros sont également étendues.

'Extras' 'Un niveau / tous les niveaux de la macro en arrière'

Icônes: 

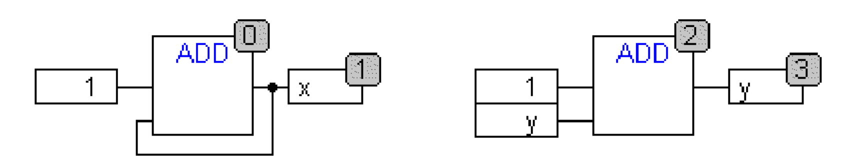
Ces commandes sont également disponibles dans la barre d'outils dès qu'une macro est ouverte pour édition. Si des macros sont imbriquées les unes dans les autres, vous pouvez revenir au niveau de macro juste au dessus ou au niveau le plus élevé.

Asservissements

Contrairement au schéma en blocs fonctionnels ordinaire, l'éditeur graphique du schéma en blocs fonctionnels permet de visualiser directement des asservissements. Il faut savoir qu'une variable intermédiaire interne est toujours créée pour la sortie d'un module. Dans le cas d'un opérateur, le type de données de la variable intermédiaire est donné par le plus grand type de données des entrées.

Le type de données d'une constante est obtenu à partir du plus petit type de données possible pour cette constante, c'est-à-dire qu'avec la constante "1" on obtient le type de données SINT. Si l'on effectue à présent une addition avec asservissement avec la constante "1", la première entrée fournit le type de données SINT et la deuxième reste indéfinie en raison de l'asservissement. Par conséquent, la variable intermédiaire est elle aussi de type SINT. La valeur de la variable intermédiaire est affectée seulement ensuite à la variable de sortie.

La figure ci-dessous montre tantôt une addition avec asservissement et tantôt une addition directe avec une variable. Dans ces cas, les variables x et y doivent être de type INT.



Les deux additions diffèrent par les points suivants:

La variable y peut être initialisée avec une valeur différente de 0, ce qui n'est pas le cas pour la variable intermédiaire de l'addition de gauche.

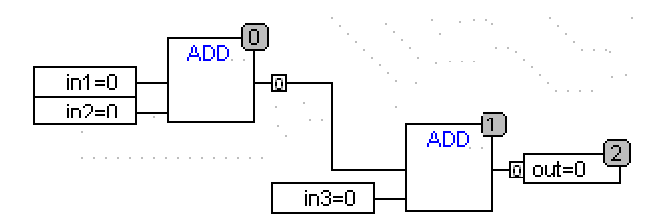
La variable intermédiaire de l'addition de gauche possède le type de données SINT tandis que la variable intermédiaire de l'addition de droite possède le type de données INT. Les variables x et y prennent des valeurs différentes à partir du 129ème appel. La variable x, bien qu'étant de type INT, prend la valeur –127, étant donné que la capacité de la variable intermédiaire est dépassée. En revanche, la variable y prend la valeur 129.

CFC en mode en ligne

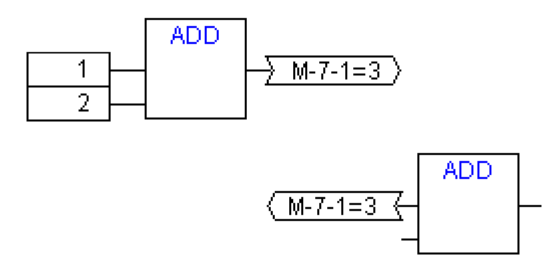
Espionnage :

Les valeurs relatives aux entrées et sorties sont représentées au sein des boîtes d'entrée et de sortie. Les constantes ne sont pas espionnées. Pour les variables non booléennes, les boîtes sont agrandies en fonction de la valeur affichée. Pour des liaisons booléennes, le nom de la variable ainsi que la liaison sont visualisés en bleu si leur valeur est TRUE, et en noir dans les autres cas.

En mode En ligne, les variables internes booléennes sont également visualisées en bleu si leur valeur est TRUE, et en noir dans les autres cas. La valeur de liaisons internes non booléennes est représentée dans une petite boîte aux coins arrondis sur le pin de sortie de la liaison.



Les pins de macros sont espionnés comme des boîtes d'entrée et de sortie.



Les liaisons non booléennes avec étiquette de liaison affichent leur valeur au sein de cette étiquette de liaison. Pour des liaisons booléennes, les liaisons ainsi que les labels sont visualisé(e)s en bleu pour autant que cette liaison soit TRUE, et en noir dans les autres cas.

Contrôle de déroulement:

Si le contrôle de déroulement est activé, les liaisons parcourues seront marquées de la couleur réglée dans les options du projet.

Points d'arrêt :

Des points d'arrêt peuvent être attribués aux éléments qui possèdent également un index d'ordre de d'exécution. L'exécution du programme est interrompue avant l'exécution même de l'élément concerné, c.-à-d. pour les modules et sorties avant l'attribution des entrées, et pour les étiquettes de saut avant l'exécution de l'élément à l'index suivant. L'index d'exécution de l'élément est utilisé comme emplacement de point d'arrêt dans la boîte de dialogue des points d'arrêt.

Le positionnement des points d'arrêt à un élément sélectionné se fait par le biais de la touche F9, de l'option 'Point d'arrêt actif/inactif' du menu 'En Ligne' ou du menu 'Extras' ou encore dans le menu contextuel de l'éditeur. Si un point d'arrêt est attribué à un élément, celui-ci sera rendu actif ou inactif par le biais de la commande 'Point d'arrêt actif/inactif'. Vous pouvez également changer la valeur d'un point d'arrêt en double-cliquant sur celui-ci.

La visualisation du point d'arrêt se fait dans la couleur réglée dans les options du projet.

Label RETURN :

En mode En ligne, une étiquette de saut est automatiquement créée avec la dénomination RETURN dans la première colonne et après le dernier élément dans l'éditeur. Cette étiquette marque la fin du module et on y passe en mode pas à pas avant de quitter le module. Des étiquettes RETURN ne sont pas insérées dans des macros.

Pas à pas :

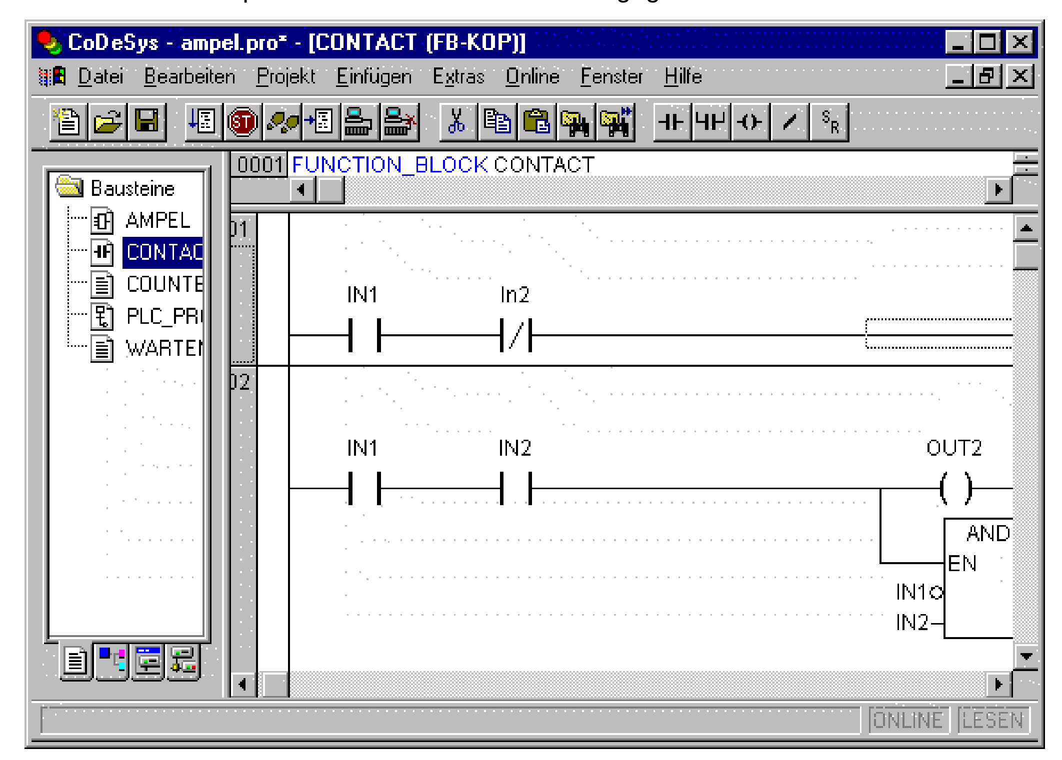
Avec 'Étape individuelle sur', on saute toujours à l'élément avec l'index d'exécution suivant (en grandeur). Si l'élément en cours est une macro ou un module, on branche vers leur implémentation au moyen de 'Cycle indépendant'. Si on exécute à partir de là une 'Étape individuelle sur', on branche à l'élément dont l'index d'exécution suit celui de la macro.

Zoom sur module appelé

Cette commande est disponible dans le menu contextuel (<F2>) ou dans le menu Extras, pour autant que le curseur se trouve sur le nom du module appelé dans les éditeurs littéraux, ou si la boîte d'un module est marquée dans les éditeurs graphiques. Le zoom ouvre le module concerné dans sa fenêtre d'éditeur. S'il s'agit d'un module issu d'une bibliothèque, alors le gestionnaire de bibliothèques est appelé et le module correspondant est affiché.

5.4.4 L'éditeur du Langage à contacts (LD)

Voici comment se présente un module écrit en langage à contacts dans l'éditeur de **CoDeSys**:



Tous les éditeurs pour modules sont constitués d'une partie de déclaration et d'un corps. Ceux-ci sont séparés entre eux par une barre de fractionnement.

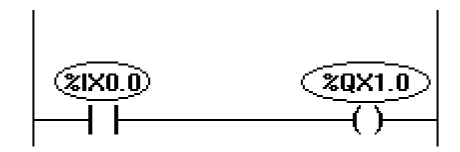
L'éditeur LD est un éditeur graphique. Vous trouvez les commandes les plus importantes dans le menu contextuel (touche droite de la souris ou <Ctrl>+<F10>).

Pour obtenir des informations sur les éléments de l'éditeur, reportez vous au chapitre 2.2.5, Langage à contacts (LD).

Positions du curseur dans l'éditeur LD

Vous trouvez ci-dessous les endroits où le curseur peut se positionner, sachant que les appels de blocs fonctionnels et de programmes peuvent être traités comme des contacts. Les modules avec entrées "EN" et les modules qui y sont rattachés sont traités comme dans le schéma en blocs fonctionnels. Vous trouvez des informations sur l'édition de ces éléments de réseau dans le chapitre 2.2.3 sur l'éditeur FBD.

1. Chaque champ de texte (la position du curseur potentielle est encadrée en noir)



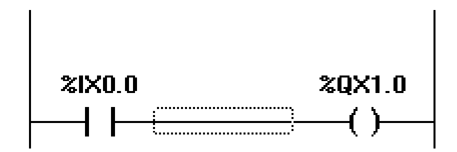
2. Chaque contact ou bloc fonctionnel



3. Chaque bobinage



4. L'élément de liaison entre les contacts et les bobinages

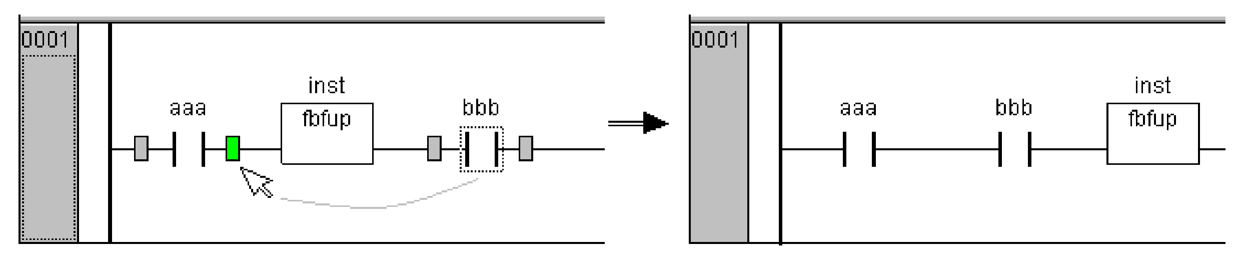


Voici les commandes de menu spécifiques au langage à contacts:

Déplacer des éléments au sein de l'éditeur de langage à contacts

Un élément dans un module LD peut être déplacé vers une autre position dans le même module grâce à la fonction "Glisser-Déplacer".

Marquez pour ce faire le contact ou le bobinage ou encore le bloc fonctionnel , puis déplacez-le de sa position actuelle tout en maintenant le bouton de la souris enfoncé. Ensuite, toutes les positions pouvant accueillir l'élément au sein du réseau de module sont alors affichées par des cases grises. Dès que l'élément est déplacé sur un de ces marquages, celui-ci est passe alors au vert. Dès que vous relâchez le bouton de la souris, l'élément est inséré à sa nouvelle position.



'Insérer' 'Contact'

Icône : **Raccourci : <Ctrl>+<O>**

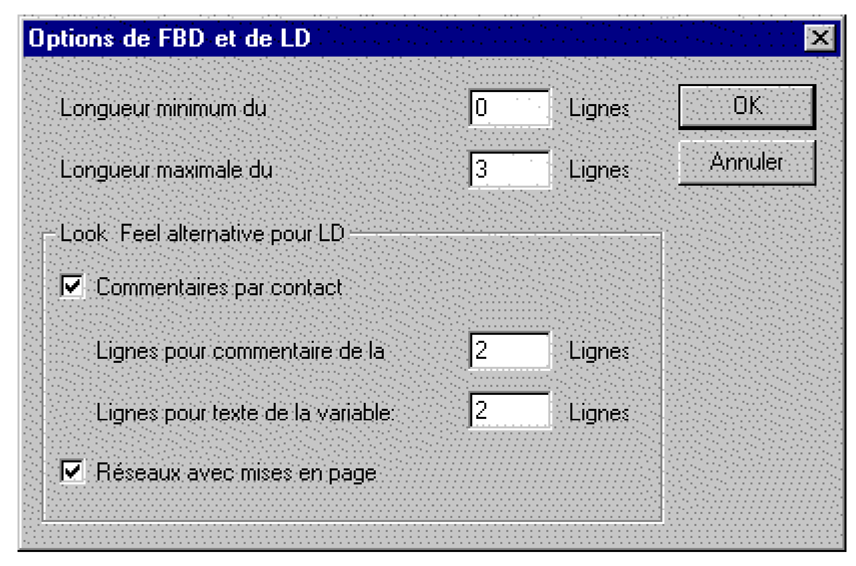
Dans l'éditeur LD, cette commande vous permet d'insérer un contact devant l'endroit sélectionné dans le réseau.

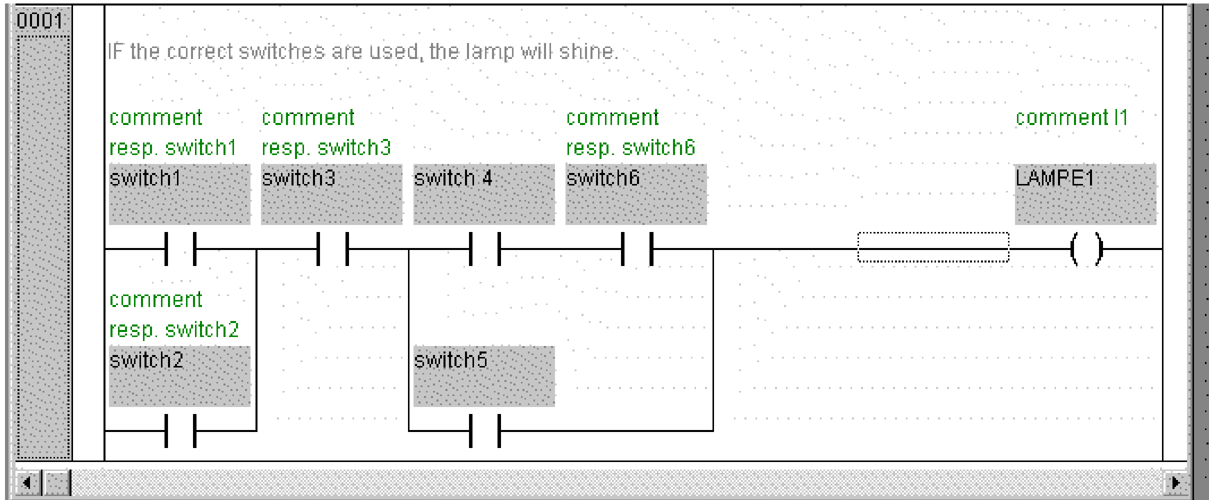
Si l'endroit sélectionné est un bobinage ou l'élément de liaison entre les contacts et les bobinages , alors le nouveau contact est monté en série sur le montage à contacts existant.

Le contact est prédéfini avec le texte "???". Vous pouvez cliquer sur ce texte et le remplacer par la variable ou la constante de votre choix. Pour ce faire, vous pouvez également utiliser la liste de sélection pour l'édition. Si vous avez activé l'option **Commentaires par contact** par le biais de 'Extras' 'Options' dans la boîte de dialogue 'Options de FBD et de LD', vous pouvez, dans cette même boîte de dialogue, définir un nombre précis de **Lignes pour texte de la variable** en plus du nombre souhaité de **Lignes pour commentaire de la variable**. Ceci est particulièrement recommandé pour des noms de variables assez longs, et permet de maintenir un réseau compact à l'horizontale.

Notez également la possibilité de l'option **Réseaux avec mise en page** que vous pouvez activer via 'Extras' 'Options'.

Exemple des options et représentation dans le réseau :





'Insérer' 'Bloc fonctionnel'

Raccourci : <Ctrl>+<B>

Vous utilisez cette commande pour insérer un bloc fonctionnel ou un programme sous forme de module. Il faut que la liaison entre les contacts et les bobinages soit marquée (position du curseur 4), ou un bobinage (position du curseur 3). La liste de sélection pour l'édition s'ouvre, vous permettant de choisir entre les modules standard disponibles ainsi que des modules que vous avez personnellement définis.

La première entrée du module fraîchement inséré est placée sur la liaison d'entrée et la première sortie est placée sur la liaison de sortie, ce qui implique que ces variables doivent toujours être de type BOOL. Toutes les autres entrées et sorties du module sont prédéfinies avec le texte "???". Ces textes prédéfinis peuvent être remplacés par des constantes, des variables ou des adresses. Pour ce faire, vous pouvez également utiliser la liste de sélection pour l'édition.

'Insérer' 'Contact parallèle'


Icône : **Raccourci : <Ctrl>+<R>**

Dans l'éditeur LD, cette commande vous permet d'insérer un contact en parallèle par rapport à l'endroit sélectionné dans le réseau.

Si l'endroit sélectionné est un bobinage ou la liaison entre les contacts et les bobinages , alors le nouveau contact est monté en parallèle sur l'ensemble du montage à contacts existant.

Le contact est prédéfini avec le texte "???". Vous pouvez cliquer sur ce texte et le remplacer par la variable ou la constante de votre choix. Pour ce faire, vous pouvez également utiliser la liste de sélection pour l'édition.

Il est possible de représenter un nom de variable sur plusieurs lignes ainsi qu'un commentaire propre pour le contact. Voyez à cet effet 'Insérer' 'Contact'

'Insérer' 'Bobinage'


Icône : Raccourci : <Ctrl>+<L>

Dans l'éditeur LD, cette commande vous permet d'insérer un bobinage monté en parallèle sur les bobinages existants.

Si l'endroit sélectionné est une liaison entre les contacts et les , alors le nouveau bobinage est inséré en dernier par rapport aux précédents. Si l'endroit sélectionné est un bobinage , alors le nouveau bobinage est inséré directement au dessus du premier.

Le bobinage est prédéfini avec le texte "???". Vous pouvez cliquer sur ce texte et le remplacer par la variable de votre choix. Pour ce faire, vous pouvez également utiliser la liste de sélection pour l'édition.

Il est possible de représenter un nom de variable sur plusieurs lignes ainsi qu'un commentaire propre pour le bobinage. Voyez à cet effet 'Insérer' 'Contact'

Modules avec entrées «EN»

Si, avec votre réseau LD, vous voulez utiliser des appels d'autres modules pour la commande, alors vous devez insérer un module avec une entrée «EN». Un tel module est monté en parallèle sur les bobinages. Avec ce module comme point de départ, vous pouvez développer le réseau comme dans le cas du schéma en bloc fonctionnels. Les commandes concernant les opérations d'insertion au niveau d'un module «EN» sont accessibles dans l'option 'Insérer' 'Ajouter au module'.

Un opérateur, un bloc fonctionnel, une fonction ou une programme doté d'une entrée «EN» se comporte comme le module correspondant dans le schéma en blocs fonctionnels, à la différence que son exécution est commandée par l'entrée «EN». Cette entrée est branchée sur la liaison entre les bobinages et les contacts. Si cet élément véhicule l'information "ON", alors le module est évalué.

Si un module doté d'une entrée «EN» a déjà été créé, il est possible de créer à partir de celui-ci un réseau semblable à ceux existants dans le schéma en blocs fonctionnels. Cela signifie qu'un module «EN» peut recevoir un flux de données des opérateurs, des fonctions ou des blocs fonctionnels courants et qu'en retour il peut transmettre des données à des modules courants de ce type.

Par conséquent, si vous voulez programmer un réseau à la façon du schéma en blocs fonctionnels dans l'éditeur LD, il vous suffit de commencer par insérer un opérateur «EN» dans un nouveau réseau, avant de développer votre réseau à partir de ce module, comme vous le feriez dans le schéma en blocs fonctionnels. Un réseau constitué de la sorte se comporte de la même façon qu'un réseau semblable dans le schéma en blocs fonctionnels.

'Insérer' 'Bloc avec EN'

Cette commande vous permet d'insérer un bloc fonctionnel, une fonction ou une programme avec entrée «EN» dans un réseau LD. Cette commande vous permet d'insérer un opérateur avec entrée «EN» dans un réseau LD.

Il faut que l'endroit sélectionné soit une liaison entre les contacts et les bobinages (position du curseur 4) ou un bobinage . Le nouvel opérateur est inséré en parallèle par rapport aux bobinages, en dessous de ceux-ci. Dans un premier temps, il est identifié par AND. Vous pouvez remplacer cette désignation par celle de votre choix. Pour ce faire, vous pouvez également utiliser la liste de sélection pour l'édition.

'Insérer' 'Ajouter au module'

Cette commande vous permet d'ajouter des éléments supplémentaires à un module inséré auparavant (il peut s'agir d'un module avec entrée «EN»). Les commandes proposées dans cette option peuvent être exécutées pour les mêmes positions du curseur que celles prévues pour les commandes correspondantes dans le schéma en blocs fonctionnels.

Entrée entraîne l'ajout d'une nouvelle entrée au module.

Sortie entraîne l'ajout d'une nouvelle sortie au module.

Vous pouvez ajouter un module supplémentaire avec **Boîte**. La procédure est la même que celle décrite sous 'Insérer' 'Module'.

Affectation vous permet d’insérer un affectation à une variable. Celle-ci est tout d’abord représentée par 3 points d’interrogation"???" que vous pouvez éditer et remplacer par la variable voulue. Vous pouvez bien entendu utiliser la liste de sélection pour l’édition.

Fonction entraîne l'ajout d'une fonction à l'entrée sélectionnée.

Bloc Fonctionnel entraîne l'ajout d'un bloc fonctionnel à l'entrée sélectionnée.

'Insérer' 'Saut'

Cette commande vous permet d'insérer un saut en parallèle et à la suite des bobinages existants, dans l'éditeur LD. Si la liaison gauche fournit comme valeur "ON", alors un saut vers l'étiquette indiquée est effectué.

Il faut que l'endroit sélectionné soit une liaison entre les contacts et les bobinages ou un bobinage .

Le saut est prédéfini avec le texte "???". Vous pouvez cliquer sur ce texte et le remplacer par l'étiquette de saut de votre choix.

'Insérer' 'Return'

Cette commande vous permet d'insérer une instruction RETURN en parallèle et à la suite des bobinages existants, dans l'éditeur LD. Si la liaison gauche fournit comme valeur "ON", alors l'exécution du module est annulée dans ce réseau.

Il faut que l'endroit sélectionné soit une liaison entre les contacts et les bobinages ou un bobinage.

'Extras' 'Insérer ci-après'

Dans l'éditeur LD, cette commande vous permet d'insérer après l'endroit sélectionné le contenu du presse-papiers, sous forme de contact monté en série. Cette commande n'est autorisée que si le contenu du presse-papiers et l'endroit sélectionné sont des réseaux à contacts.

'Extras' 'Insérer ci-dessous'

Raccourci : <Ctrl>+<U>

Dans l'éditeur LD, cette commande vous permet d'insérer en dessous de l'endroit sélectionné le contenu du presse-papiers, sous forme de contact monté en parallèle. Cette commande n'est autorisée que si le contenu du presse-papiers et l'endroit sélectionné sont des réseaux à contacts.

'Extras' 'Insérer ci-dessus'

Dans l'éditeur LD, cette commande vous permet d'insérer au dessus de l'endroit sélectionné le contenu du presse-papiers, sous forme de contact monté en parallèle. Cette commande n'est autorisée que si le contenu du presse-papiers et l'endroit sélectionné sont des réseaux à contacts.

'Extras' 'Négation'

Icône : **Raccourci : <Ctrl>+<N>**

Cette commande vous permet d'inverser un contact, un bobinage, un saut ou une instruction RETURN, ou encore l'entrée ou la sortie de modules «EN» au niveau de la position actuelle du curseur .

Entre les parenthèses du bobinage ou entre les traits droits du contact apparaît une barre oblique (respectivement (/) et |/|). Dans le cas de sauts, d'instructions Return, d'entrées ou de sorties de modules «EN», un petit cercle apparaît sur la liaison, comme pour l'éditeur FBD.

Le bobinage écrit à présent la valeur inverse de la liaison d'entrée dans la variable booléenne correspondante. Un contact inversé fait commuter l'état de l'entrée sur la sortie au moment où la variable booléenne correspondante fournit comme valeur FALSE.

Si un saut ou une instruction Return est marquée, alors la valeur de l'entrée de ce saut ou de cette instruction, selon le cas, est inversée.

Une négation peut être annulée en inversant la valeur une nouvelle fois.

'Extras' 'Set/Reset'

Si vous exécutez cette commande pour un bobinage, alors vous obtenez un bobinage SET. Un tel bobinage n'écrase jamais la valeur TRUE contenue dans la variable booléenne correspondante. Autrement dit, si la valeur TRUE est affectée à cette variable, alors elle le sera de façon permanente. Un bobinage SET est caractérisé par un "S" au niveau du symbole de bobinage.

Si vous répétez cette commande, alors vous obtenez un bobinage RESET. Un tel bobinage n'écrase jamais la valeur FALSE contenue dans la variable booléenne correspondante. Autrement dit, si la valeur FALSE est affectée à cette variable, alors elle le sera de façon permanente. Un bobinage RESET est caractérisé par un "R" au niveau du symbole de bobinage.

Si vous répétez plusieurs fois cette commande, alors le bobinage devient tour à tour un bobinage SET, un bobinage RESET et un bobinage normal.

Couper, copier, coller et supprimer dans le schéma en blocs fonctionnels

Les commandes '**Couper**', '**Copier**', '**Coller**' et '**Supprimer**' se trouvent dans le menu 'Editer'.

Si une jonction est sélectionnée , alors les affectations, sauts ou instructions RETURN situés en dessous de celle-ci sont coupés, supprimés ou copiés.

Si une boîte est sélectionnée, l'objet sélectionné est coupé, supprimé ou copié, ainsi que toutes les branches contiguës aux entrées, à l'exception de la première (plus élevée) d'entre elles.

Dans les autres cas, la totalité de la branche située devant la position du curseur est coupée, supprimée ou copiée.

Après l'opération de copiage ou de découpage, l'élément supprimé ou copié se trouve dans le presse-papiers et peut être collé autant de fois que l'on veut.

Pour ce faire, il faut d'abord sélectionner la position d'insertion. Les entrées et les sorties sont des positions d'insertion possibles.

Si un boîte a été chargé dans le presse-papiers, (pour mémoire: dans ce cas, toutes les branches contiguës se trouvent dans le presse-papiers, à l'exception de la première), alors la première entrée est reliée à la branche située devant la position d'insertion.

Dans le cas contraire, la totalité de la branche située devant la position d'insertion est remplacée par le contenu du presse-papiers.

Dans tous les cas, le dernier élément collé est relié à la branche située à droite de la position d'insertion.

Remarque : Le découpage et le collage permettent de résoudre le problème suivant. Supposons qu'un opérateur soit collé au milieu d'un réseau. La branche située à droite de l'opérateur est à présent reliée à la première entrée, alors qu'elle devrait être reliée à la deuxième entrée. On sélectionne à présent la première entrée et on exécute la commande 'Editer' 'Couper'. Ensuite, on sélectionne la deuxième entrée et on exécute la commande 'Editer' '**Coller**'. La branche est maintenant rattachée à la deuxième entrée.

Zoom sur module appelé

Cette commande est disponible dans le menu contextuel (<F2>) ou dans le menu Extras, pour autant que le curseur se trouve sur le nom du module appelé dans les éditeurs littéraux, ou si la boîte d'un module est marquée dans les éditeurs graphiques. Le zoom ouvre le module concerné dans sa fenêtre d'éditeur. S'il s'agit d'un module issu d'une bibliothèque, alors le gestionnaire de bibliothèques est appelé et le module correspondant est affiché.

Le langage à contacts en mode En Ligne

Lorsqu'on utilise le langage à contacts dans le mode En Ligne, tous les contacts et bobinages à l'état "ON" sont colorés en bleu, de même que toutes les liaisons qui véhiculent l'information "ON". Les valeurs des variables correspondantes sont visualisées au niveau des entrées et sorties des blocs fonctionnels.

Les points d'arrêt peuvent être définis uniquement au niveau des réseaux; en pas à pas, la progression s'effectue en sautant de réseau en réseau.

Si vous maintenez le pointeur de souris pendant une courte durée sur une variable, le type et le commentaire de la variable sont affichés dans une info-bulle.

5.4.5 L'éditeur du Diagramme fonctionnel en séquence (SFC)

Voici comment se présente un module écrit en SFC dans l'éditeur **CoDeSys**:

Tous les éditeurs pour modules sont constitués d'une partie de déclaration et d'un corps. Ceux-ci sont séparés entre eux par une barre de fractionnement.

L'éditeur SFC est un éditeur graphique. Vous trouvez les commandes les plus importantes dans le menu contextuel (touche droite de la souris ou <Ctrl>+<F10>).

Des info-bulles vous indiquent les noms complets ou expressions d'étapes, transitions, sauts, étiquettes, qualificatifs ou actions associées, que ce soit en mode En ou Hors ligne ou même en zoom.

Pour obtenir des informations sur le diagramme fonctionnel en séquence, reportez vous au chapitre 2.2.2, Diagramme fonctionnel en séquence (SFC).

Sélectionner des blocs en SFC

Un bloc sélectionné est un ensemble d'éléments SFC entourés d'un rectangle en pointillés

Un élément (une étape, une transition ou un saut) peut être sélectionné en positionnant le pointeur de la souris dessus et en appuyant sur le bouton gauche de la souris, ou encore en utilisant les touches directionnelles. Pour sélectionner un ensemble d'éléments, appuyez sur la touche <Maj> après avoir sélectionné un bloc et sélectionnez l'élément situé dans le coin inférieur gauche ou droit de l'ensemble. La sélection obtenue est le plus petit ensemble d'éléments contigus qui contient ces deux éléments.

Veuillez tenir compte du fait que toutes les commandes seront exécutées seulement si elles sont compatibles avec les conventions du langage.

Veuillez noter que vous ne pouvez effacer une étape qu'avec la transition précédente ou suivante.

'Insérer' 'Etape-Transition (devant)'

Icône : **Raccourci : <Ctrl>+<T>**

Dans l'éditeur SFC, cette commande permet d'insérer devant le bloc sélectionné une étape suivie d'une transition.

'Insérer' 'Etape-Transition (derrière)'

Icône : **Raccourci : <Ctrl>+<E>**

Dans l'éditeur SFC, cette commande permet d'insérer derrière le bloc sélectionné une étape suivie d'une transition.

Effacer une étape et une transition

Vous ne pouvez effacer une étape qu'avec la transition précédente ou suivante. Veuillez marquer à cet effet l'étape et la transition et exécutez la commande 'Editer' 'Supprimer' ou appuyez sur la touche <Suppr>.

'Insérer' 'Séquence alternative (droite)'

Icône : Raccourci : <Ctrl>+<A>

Dans l'éditeur SFC, cette commande ajoute au bloc sélectionné une séquence alternative vers la droite. Pour que cela soit possible, le bloc sélectionné doit commencer et finir par une transition. La nouvelle branche est constituée alors d'une transition.

'Insérer' 'Séquence alternative (gauche)'

Icône : 

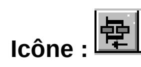
Dans l'éditeur SFC, cette commande ajoute au bloc sélectionné une séquence alternative vers la gauche. Pour que cela soit possible, le bloc sélectionné doit commencer et finir par une transition. La nouvelle branche est constituée alors d'une transition.

'Insérer' 'Séquence parallèle' (droite)'

Icône : Raccourci : <Ctrl>+<L>

Dans l'éditeur SFC, cette commande ajoute au bloc sélectionné une séquence parallèle vers la droite. Pour que cela soit possible, le bloc sélectionné doit commencer et finir par une étape. La nouvelle branche est constituée alors d'une étape. Pour permettre des sauts vers la séquence parallèle créée, elle doit être munie d'une étiquette de saut. (voir 'Extras' 'Ajouter une étiquette à la branche parallèle')

'Insérer' 'Séquence parallèle (gauche)"

Icône : 

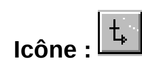
Dans l'éditeur SFC, cette commande ajoute au bloc sélectionné une séquence parallèle vers la gauche. Pour que cela soit possible, le bloc sélectionné doit commencer et finir par une étape. La nouvelle branche est constituée alors d'une étape. Pour permettre des sauts vers la séquence parallèle créée, elle doit être munie d'une étiquette de saut. (voir 'Extras' 'Ajouter une étiquette à la branche parallèle')

'Insérer' 'Saut'

Icône : Raccourci : <Ctrl>+<U>

Dans l'éditeur SFC, cette commande insère un saut à la fin de la branche qui se rapporte au bloc sélectionné. Pour que cela soit possible, la branche doit être une séquence alternative. Lorsqu'un saut a été inséré, il est possible de sélectionner ensuite le texte existant "Step" et de le remplacer par le nom de l'étape correspondant au saut ou par l'étiquette de saut d'une branche parallèle'.

'Insérer' 'Transition-Saut'

Icône : 

Dans l'éditeur SFC, cette commande insère une transition suivie d'un saut à la fin de la séquence sélectionnée, qui doit obligatoirement être de type parallèle. Lorsqu'un saut a été inséré, il est possible de sélectionner ensuite le texte existant "Step" et de le remplacer par le nom de l'étape correspondant au saut ou par l'étiquette de saut d'une branche parallèle'.

'Insérer' 'Ajouter une action d'entrée'

Cette commande vous permet d'ajouter une action d'entrée à une étape. Une action d'entrée est exécutée une seule fois, immédiatement après que l'étape ait été activée. L'action d'entrée peut être implémentée dans le langage de votre choix.

Une étape avec action d'entrée est caractérisée par un 'E' dans le coin inférieur gauche.

Une action d'entrée ne peut pas être définie en tant qu'étape CEI.

'Insérer' 'Ajouter une action de sortie'

Cette commande vous permet d'ajouter une action de sortie à une étape. Une action de sortie est exécutée une seule fois, avant que l'étape ne soit désactivée. L'action de sortie peut être implémentée dans le langage de votre choix.

Une étape avec action de sortie est caractérisée par un 'X' dans le coin inférieur droit.

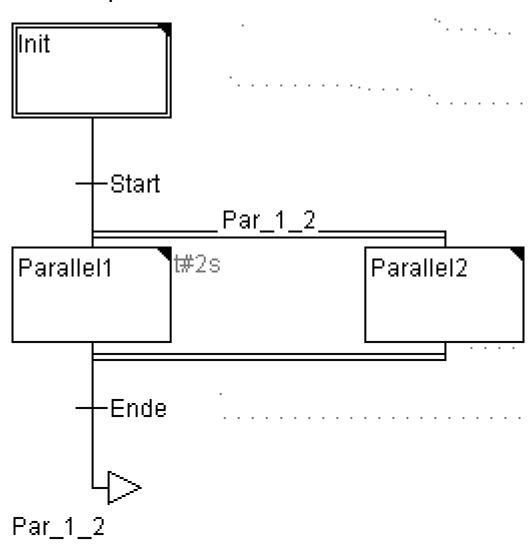
Une action de sortie ne peut pas être définie en tant qu'étape CEI.

'Extras' 'Insérer branche parallele (droite)

Cette commande insère le contenue du presse-papiers comme une séquence parallèle vers la droite du bloc sélectionné. Pour que cela soit possible, le bloc sélectionné doit commencer et finir par une étape. Le contenue du presse-papiers aussi doit être un bloc SFC, qui commence et fini par un étape.

'Extras' 'Ajouter une étiquette à la branche parallele'

Afin d'ajouter une étiquette de saut à la séquence parallèle qui vient d'être insérée, la transition se trouvant devant cette séquence parallèle doit être marquée et vous devez ensuite exécuter la commande 'Ajouter une étiquette à la branche parallèle (gauche)". Suite à quoi la séquence parallèle se voit attribuer le nom standard "Parallèle" suivi d'un numéro courant, ces deux éléments pouvant être édités selon les règles de dénomination d'identificateurs. Dans l'exemple qui suit, "Parallel" a été remplacé par "Par_1_2" et le saut placé à la fin du diagramme après la transition "Ende" branche vers cette étiquette.



Effacer une étiquette d'un saut

Vous effacez une étiquette de saut en effaçant le texte de cette même étiquette de saut.

'Extras' 'Insérer derrière'

Cette commande insère le bloc SFC stocké dans le presse-papiers après la première étape ou transition du bloc sélectionné (un copiage normal réalise une insertion avant le bloc sélectionné). Cette opération n'est exécutée que si la structure SFC résultante satisfait aux normes de langage.

'Extras' 'Zoom action/transition'

Raccourci : <Alt>+<Entrée>

L'action de la première étape ou le corps de transition de la première transition du bloc sélectionné est chargé(e) dans l'éditeur, dans le langage dans lequel il (elle) est écrit(e). Si l'action ou le corps de transition est vide, alors il faut sélectionner le langage dans lequel il (elle) doit être écrit(e).

'Extras' 'Effacer action/transition'

Cette commande vous permet d'effacer les actions de la première étape ou le corps de la première transition du bloc sélectionné.

Si, au niveau d'une étape, seule l'action, l'action d'entrée ou l'action de sortie est implémentée, alors elle sera effacée par cette commande. Dans les autres cas, une boîte de dialogue apparaît et permet de choisir quelle(s) action(s) doi(ven)t être effacée(s).

Si le curseur est positionné sur une action reliée à une étape CEI, alors uniquement cette association sera effacée par cette commande. Si une étape CEI est sélectionnée et qu'une action est reliée à cette étape, alors cette association est effacée. Dans le cas d'une étape CEI comportant plusieurs actions, une boîte de dialogue apparaît pour opérer une sélection.

'Extras' 'Attributs d'étape'

Cette commande entraîne l'ouverture d'une boîte de dialogue, dans laquelle vous pouvez éditer des attributs de l'étape sélectionnée.

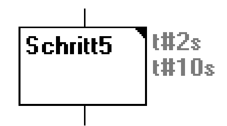
Boîte de dialogue Attributs de l'étape

Attributs de l'étape
Temps minimal: t#2ms
Temps maximal: t#10ms
OK
Annuler
Commentaire:
Duration phase 1

Vous pouvez effectuer trois entrées différentes dans la boîte de dialogue Attributs de l'étape. Dans **Temps minimal**, vous entrez le temps minimal prescrit pour l'exécution de cette étape. Dans **Temps maximal**, vous entrez le temps maximal prescrit pour l'exécution de cette étape. Nous vous signalons que les entrées doivent être du type **TIME**. Les règles d'écriture décrites dans l'annexe C s'appliquent donc au cas présent.

Dans **Commentaire** vous pouvez entrer un commentaire relatif à l'étape. Avec 'Extras' 'Options' vous pouvez configurer l'éditeur SFC de façon à visualiser au niveau de vos étapes, soit les commentaires, soit le réglage de temps. A droite de l'étape apparaît alors le commentaire ou les réglages de temps, selon le cas.

Dans le cas d'un dépassement du temps maximal, des drapeaux SFC sont mis. Ils peuvent être contrôlés par l'utilisateur.

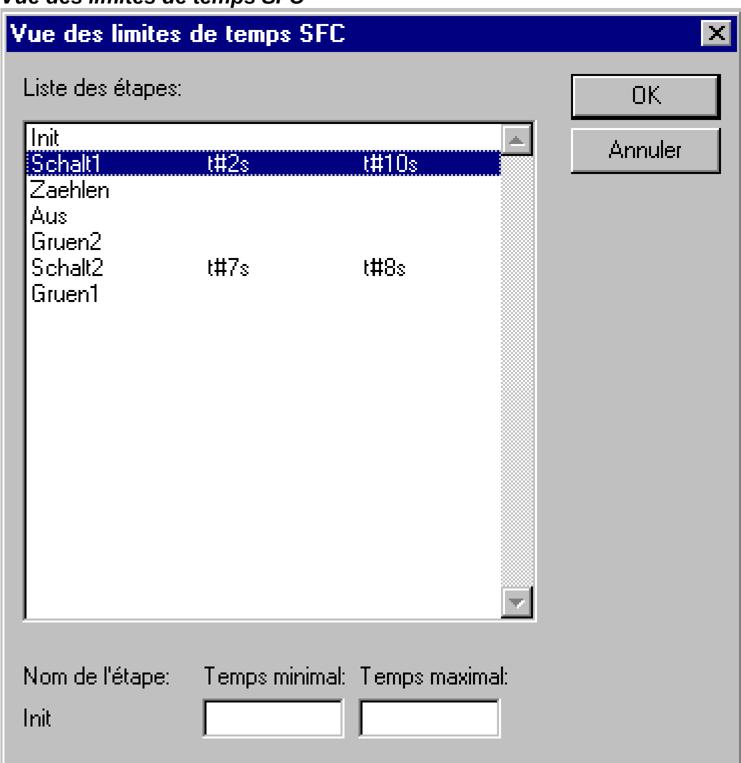


L'exemple présente le cas d'une étape dont l'exécution doit durer au moins 2 secondes et au plus 10 secondes. En mode En Ligne, l'affichage indique depuis combien de temps l'étape est active, en plus des deux autres données.

'Extras' 'Vue des temps'

Cette commande vous permet d'ouvrir une fenêtre, au sein de laquelle vous pouvez éditer les réglages de temps de vos étapes SFC:

Vue des limites de temps SFC



Dans la Vue des limites de temps SFC, toutes les étapes de votre module SFC sont visualisées. Si vous avez spécifié une limite de temps pour une étape donnée, alors celle-ci est affichée à droite de l'étape (d'abord la limite inférieure et puis la limite supérieure). En outre, vous pouvez éditer les limites de temps. Pour ce faire, cliquez sur l'étape souhaitée dans la Liste des étapes. Le **Nom de l'étape** s'affiche alors dans le bas de la fenêtre. Allez ensuite sur le champ **Temps minimal** ou **Temps maximal** et entrez y la limite de temps de votre choix. Lorsque vous fermez la fenêtre par **OK**, toutes les modifications sont enregistrées.

Dans l'exemple, les étapes 2 et 6 possèdent une limite de temps. Switch1 dure au moins deux secondes et au plus dix secondes. Switch2 dure au moins sept secondes et au plus huit secondes.

'Extras' 'Options'

Cette commande vous donne accès à une boîte de dialogue, qui vous permet de configurer différentes options pour votre module SFC.

Boîte de dialogue Options SFC

Options SFC
Hauteur de l'étape: 4 Lignes
OK
Annuler
Largeur de l'étape: 6
Largeur de Commentaire: 6
Affichage pendant l'étape
Rien
Commentaire
Contrôle du temps

Dans la boîte de dialogue Options SFC, vous pouvez effectuer cinq entrées. Dans **Hauteur de l'étape** vous pouvez entrer le nombre de lignes requises pour une étape SFC, au sein de l'éditeur SFC. La configuration standard est 4. Dans **Largeur de l'étape** vous pouvez entrer le nombre de colonnes requises pour une étape. La configuration standard est 6. La **Largeur de commentaire** définit le nombre de caractères affichés lorsque vous laissez afficher le commentaire avec l'étape.

Dans la zone **Affichage pendant l'étape** vous pouvez sélectionner les entrées faites dans 'Extras' 'Attributs d'étape' qui doivent être affichées. Vous pouvez opter soit pour **Rien**, soit pour Commentaire, soit pour **Contrôle de temps**.

'Extras' 'Relier action'

Cette commande permet d'associer des actions et des variables booléennes à des étapes CEI.

Une petite boîte en deux parties est ajoutée à côté de l'étape CEI, pour permettre d'associer une action. Le qualificateur 'N' et le nom 'Action' sont prédéfinis dans la partie gauche de la boîte. Ces deux préaffectations peuvent être modifiées. Pour ce faire, vous pouvez utiliser la liste de sélection pour l'édition.

Vous pouvez affecter un maximum de neuf actions à une étape CEI.

De nouvelles actions pour des étapes CEI peuvent être créées dans l'Organisateur d'objets au niveau d'un module SFC, au moyen de la commande 'Projet' 'Ajouter une action'.

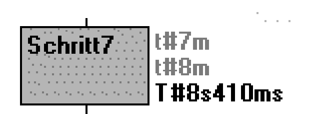
'Extras' 'Utiliser les étapes CEI'

Icône : 

Si cette commande est activée (dans ce cas un crochet (✓) apparaît devant l'élément de menu et l'icône correspondante de la barre de fonction est enfoncée), alors des étapes CEI, plutôt que des étapes simplifiées, sont insérées lors de l'insertion de transitions d'étapes et de séquences parallèle.

Le diagramme fonctionnel en séquence mode En Ligne

Dans le cas où l'éditeur du diagramme fonctionnel en séquence opère En Ligne, les étapes actuellement actives sont visualisées en bleu (en noir dans l'exemple). Si vous avez effectué la configuration appropriée dans 'Extras' 'Options', alors le contrôle du temps est visualisé à côté des étapes. En dessous des limites inférieures et supérieures de temps que vous avez spécifiées, une troisième donnée temporelle apparaît, vous indiquant depuis combien de temps l'étape est active.



Dans l'illustration ci-dessus, l'étape représentée est active depuis 8 secondes et 410 millisecondes. Cette étape doit toutefois demeurer active pendant 7 minutes avant qu'elle ne soit abandonnée.

Il est possible de définir un point d'arrêt au niveau d'une étape au moyen de 'En Ligne' 'Point d'arrêt actif/inactif'., et au niveau d'une action aux endroits d'arrêt permis par le langage. Le traitement est suspendu avant même l'exécution de cette étape ou de l'endroit du programme de cette action. Les étapes ou endroits de programme sur lesquel(le)s un point d'arrêt est défini sont marqué(e)s en bleu clair.

Si plusieurs étapes sont actives dans une séquence parallèle, alors l'étape dont l'action sera la prochaine à être traitée est visualisée en rouge.

Si des étapes CEI ont été utilisées, alors toutes les actions actives sont visualisés en bleu dans le mode En Ligne.

Le diagramme fonctionnel en séquence supporte lui aussi le pas à pas ('En Ligne' 'Etape individuelle sur'). Dans ce cas, on progresse jusqu'à la prochaine étape dont l'action sera exécutée.

Vous pouvez atteindre directement une étape, que ce soit une action ou une transition, au moyen de 'En Ligne' 'Etape individuelle dans'.

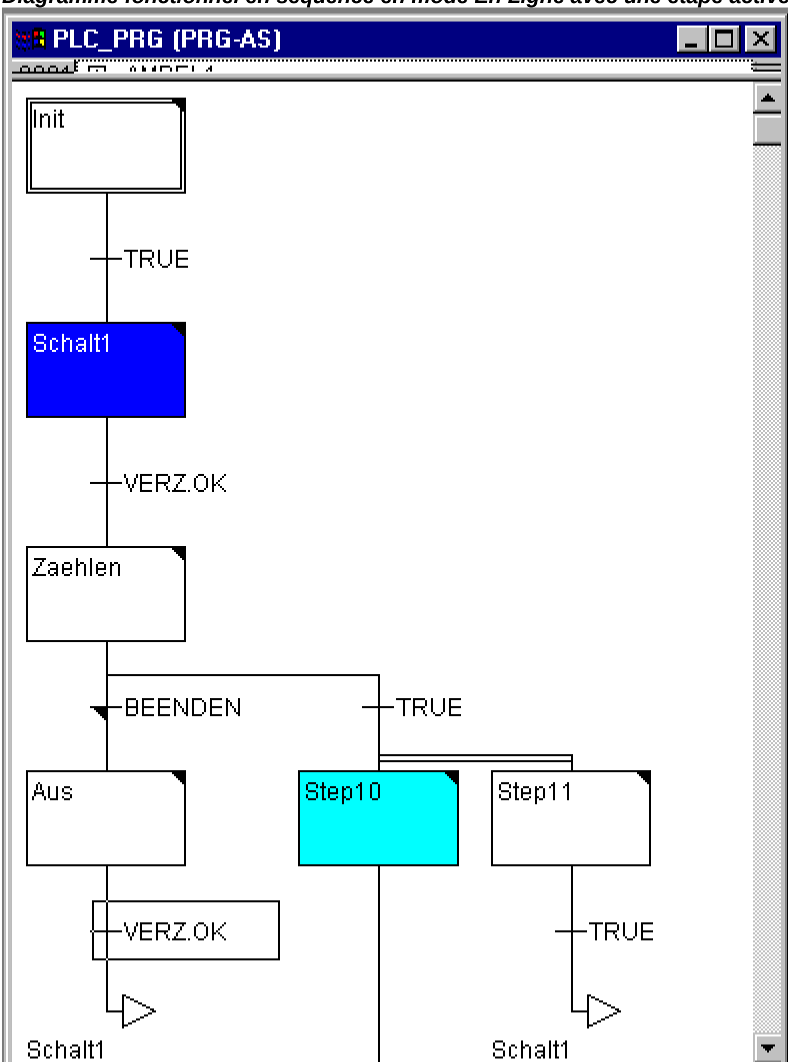
La commande **'En Ligne' 'Étape individuelle sur'** permet de passer pas à pas à l'étape suivante dont l'action est exécutée. Si la position actuelle est :

- une étape dans un déroulement linéaire d'un module ou encore une étape dans la séquence parallèle la plus à droite d'un module, on quitte le module SFC et on revient à l'appelant. Si le module est le programme principal, le cycle suivant débute.
- une étape dans la séquence parallèle n'étant pas la plus à droite, on saute à l'étape active de la séquence parallèle suivante.
- le dernier point d'arrêt au sein d'une action 3S, on saute vers l'appelant du SFC.
- le dernier point d'arrêt au sein d'une action CEI, on saute vers l'appelant du SFC.
- le dernier point d'arrêt au sein d'une action d'entrée ou de sortie, on saute vers la première étape active.

Vous pouvez en outre avancer pas à pas dans des actions au moyen de "**En Ligne**" + "**Étape individuelle dans**". Si vous devez sauter vers une action d'entrée, de sortie ou CEI, vous devez y créer un point d'arrêt. A l'intérieur des actions, l'utilisateur dispose de toutes les fonctionnalités de débogage de l'éditeur correspondant.

Si vous maintenez le pointeur de souris pendant une courte durée sur une variable dans l'éditeur de déclaration, le type et le commentaire de la variable sont affichés dans une info-bulle.

Diagramme fonctionnel en séquence en mode En Ligne avec une étape active (Switch1) et un point d'arrêt (Step 10)



Veuillez noter : Si vous renommez une étape et effectuez un changement En ligne alors que cette même étape est en cours d'exécution, le programme s'arrête avec un état indéfini.

Ordre d'exécution des éléments d'une chaîne d'étapes :

1. Tous les drapeaux du bloc de contrôle des actions CEI utilisées dans cette chaîne sont tout d'abord remis à zéro. (ne concerne cependant pas les drapeaux d'actions CEI appelées à l'intérieur des actions).
2. Pour chaque action et dans leur ordre d'apparition dans la chaîne (du haut vers le bas et de gauche à droite), on vérifie si la condition pour l'exécution de l'action de sortie est présente et réalisée le cas échéant.
3. Pour chaque action et dans leur ordre d'apparition dans la chaîne, on vérifie si la condition pour l'exécution de l'action d'entrée est présente et réalisée le cas échéant.
4. Pour chaque étape et dans leur ordre d'apparition dans la chaîne, les points suivants sont exécutés :
   - le temps écoulé est éventuellement copié dans la variable d'étape correspondante.
   - on vérifie éventuellement un dépassement de temps et on utilise les drapeaux d'erreur SFC correspondants.
   - pour les actions autres que CEI, l'action correspondante est exécutée.
5. Les actions CEI utilisées dans la chaîne sont exécutées dans l'ordre alphabétique. À cet effet, la liste des actions est parcourue deux fois. Lors du premier passage, toutes les actions CEI désactivées dans le cycle en cours sont exécutées. Lors du second passage, toutes les actions CEI actives dans le cycle en cours sont exécutées.
6. Les transitions sont évaluées : si l'étape dans le cycle en cours était active et que la transition y faisant suite donne la valeur TRUE (et que le temps minimal est éventuellement déjà écoulé), l'étape suivante est activée.

Observez les points suivants lors de l'implémentation d'actions :

Il se peut qu'une action soit exécutée plusieurs fois au cours d'un cycle, du fait qu'elle soit associée à plusieurs chaînes. (Par exemple, un programme SFC peut contenir deux actions CEI A et B, ces deux actions étant implémentées en langage SFC et appelant toutes deux une action CEI C : au cours du même cycle, les actions CEI A et B pourraient être actives et au sein de ces deux actions, l'action CEI C pourrait également être active, ce qui a pour effet que cette dernière action C est exécutée deux fois.)

Si la même action CEI est utilisée simultanément à deux niveaux différents au sein d'un SFC, ceci pourrait avoir des effet indésirables en raison de l'ordre d'exécution décrit ci-dessus. Dans ce cas, un message d'erreur apparaît à l'éccran. Ceci peut se produire lors du traitement de projet qui ont été créés à l'aide de versions antérieures de **CoDeSys** !

Remarque : Lors de l'espionnage d'expressions (p.ex. A ET B) dans les transitions, la valeur globale de la transition est seule représentée.

6 Ressources

6.1 Aperçu du Ressources

Dans l'onglet **Ressources** de l'Organisateur d'objets, vous trouvez des objets qui permettent de configurer et d'organiser votre projet, de suivre l'évolution de la valeur des variables et de configurer le projet pour utilisation sur le système cible et dans le réseau, à savoir :

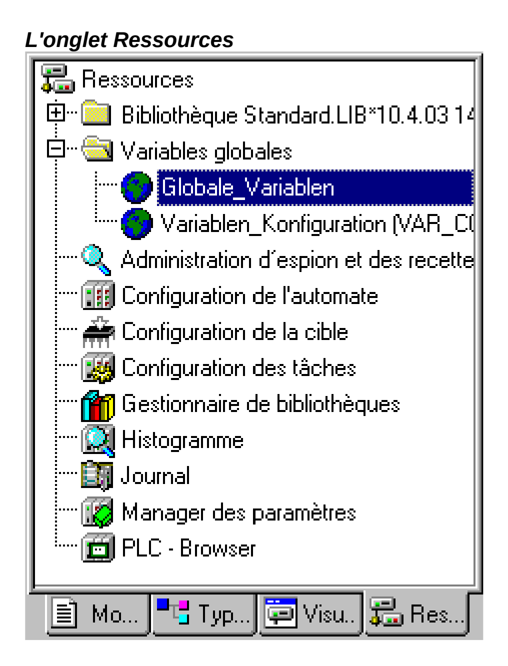
- de **Variables globales**, qui peuvent être utilisées dans l'ensemble du projet ou du réseau
- la **Configuration de l'alarme** pour configurer des classes et groupes d'alarme, qui par example peuvent visualisée avec la visualisation de CoDeSys
- la **Gestionnaire de bibliothèques** pour l'administration des bibliotheques dans un projet de la Configuration de l'automate , pour configurer votre matériel de la Configuration des tâches, pour commander votre programme au moyen de tâches
- du **Journal** pour enregistrer les actions qui ont lieu lors d'une session
- du **Gestionnaire d'espion et des recettes**, pour visualiser des valeurs de variables et préaffecter des valeurs aux variables.
- de **Configuration de la cible** permettant le choix et le cas échéant le paramétrage du matériel cible
- **Environnement de travail** avec représentation des options du projet

Dependant de la configuration de la cible en outre ce sont disponibles les ressources suivantes:

- L'**enregistrement des Histogramme**, pour l'enregistrement graphique des valeurs des variables
- Le **Manager des paramètres** qui met à disposition des variables auxquelles d'autres utilisateurs au sein du réseau peuvent accéder (Fonctionnalité dépendant de la configuration du système cible)
- Le **PLC-Browser** permettant l'appel d'information de l'automate en cours d'exécution
- Les **Outils** pour la connexion à des outils externes pouvant être démarrés à partir de CoDeSys. (La fonctionnalité dépend du système cible)
- La fonction **SoftMotion** (soumise à licence) avec l'éditeur CNC (liste de programme CNC) et l'éditeur CAM (cames)

En outre, si un objet provenant des variables globales est ouvert, un **document modèle** peut être créé et appelé et à l'aide duquel différents commentaires peuvent être mis à disposition pour une variable. Ces commentaires se retrouvent dans la documentationEn outre, si un objet provenant des variables globales est ouvert, un document modèle peut être créé et appelé et à l'aide duquel différents commentaires peuvent être mis à disposition pour une variable. Ces commentaires se retrouvent dans la documentation

L'onglet Ressources



6.2 Variables Globales, Variable Configuration, Fichier cadre pour documentation

Objets in 'Variables Globales'

A l'intérieur de l'Organisateur d'objets, dans l'onglet **Ressources**, le dossier **Variables globales** contient dans l'état standard trois objets (entre parenthèses figurent les noms prédéfinis des objets):

- Liste des variables globales (Global_Variables)
- Configuration des variables (Variable_Configuration)

Chacune des variables définies à l'intérieur de ces objets sont connues dans l'ensemble du projet, les variables réseau globales peuvent en outre servir à l'échange de données avec d'autres utilisateurs du réseau.

Si vous trouvez le dossier Variables globales fermé (un signe plus précède le dossier), ouvrez-le en double-cliquant ou en appuyant sur la touche <Entrée> au niveau de la ligne.

Choisissez l'objet approprié. La commande 'Editer objet' vous permet d'accéder à une fenêtre affichant les variables globales qui sont déjà définies. L'éditeur utilisé dans le cas présent travaille de la même manière que l'éditeur de déclaration.

Plusieurs listes de variables

Les variables globales ordinaires (**VAR_GLOBAL)** et les configurations de variable (**VAR_CONFIG**) doivent être définies dans des objets séparés.

Si vous avez déclaré un grand nombre de variables globales, et si vous voulez mieux structurer votre liste de variables globales, alors vous pouvez créer des listes de variables supplémentaires.

Sélectionnez dans l'Organisateur d'objets le dossier **Variables globales** ou un des objets présents contenant des variables globales et exécutez la commande 'Projet' 'Insérer objet'. Attribuez un nom parlant à l'objet, au moyen de la boîte de dialogue qui s'affiche. Cela entraîne la création d'un objet supplémentaire, dont le mot clé est **VAR_GLOBAL**. Si vous préférez obtenir un objet avec des variables accès ou une configuration de variables, changez respectivement le mot clé en **VAR_ACCESS** ou en **VAR_CONFIG**.

6.2.1 Variables Globales

Des variables "normales", des constantes ou des variables rémanentes peuvent être déclarées comme variables globales qui seront alors connues dans le projet global, de même que les variables réseau qui permettent en outre l'échange de données avec d'autres utilisateurs du réseau.

Remarque: Il est possible de définir une variable locale avec le même nom qu'une variable globale. Au sein d'un module, la variable définie localement a toujours priorité.
Il n'est pas possible d'attribuer le même nom à deux variables globales ; cela provoque par exemple une erreur de compilation si une variable "var1" est définie aussi bien dans la configuration de l'automate que dans une liste de variables globales.

Variables réseau

Remarque: L'utilisation de variables réseau doit être supportée par le système cible et être activée au sein de la configuration du système cible (catégorie Fonctions réseau).

Les variables réseau peuvent être maintenues au même niveau auprès de plusieurs automates par le biais d'un **échange automatique de données** et ce au sein d'un réseau d'automates compatible avec CoDeSys (comparez à ce sujet l'échange non-automatique par le biais du gestionnaire des paramètres). Ceci ne nécessite pas de fonctions spécifiques à l'automate, mais les utilisateurs réseau doivent cependant disposer dans leurs projets des mêmes listes de déclaration et de la même configuration relative à la transmission de variables réseau. Afin d'obtenir des listes identiques, nous vous recommandons de ne pas introduire manuellement la déclaration de ces variables sur chaque automate, mais bien de les reprendre d'un fichier séparé pouvant être créé par le biais d'une exportation. (Voir 'Création d'une liste de variables globales').

ATTENTION ! Pour les variables réseau globales, il n'est actuellement pas possible d'effectuer des changements En ligne. Les changements dans la configuration des variables réseau ne sont pas reconnus comme des modifications du projet ! (Défaut Id 3320)

Création d'une 'Liste de variables globales'

Pour créer une liste de variables globales, marquez l'entrée 'Variables globales' dans l'Organisateur d'objets des Ressources, ou encore une liste de variables globales qui y serait déjà créée.

Pour créer une liste de variables globales, marquez l'entrée 'Variables globales' dans l'Organisateur d'objets dans la rubrique 'Ressources', ou encore une liste de variables globales qui y serait déjà créée. Si vous choisissez ensuite la commande **'Projet' 'Objet' 'Insérer'**, la boîte de dialogue '**Propriétés**' s'ouvre pour la **Liste de variables globales**.

Cette boîte de dialogue s'ouvre par ailleurs aussi par le biais de la commande **'Projet' 'Objet' 'Propriétés'** pour afficher la configuration de la liste de variables globales dans l'organisateur d'objets, cette liste étant déjà configurée.

Boîte de dialogue de création d'une nouvelle liste de variables globales

Propriétés

Liste de variables globales

Nom de la liste de variables globales: Variables_Globales

Liaison au fichier

Nom du fichier: nwvar_1.exp — Balayer...

Importer avant la compilation — Exporter avant la compilation

Ajouter liaison au réseau

Connection 1 (UDP)

Type de réseau: UDP — Environnement...

Comprimer les variables

Identificateur de la liste de variables (COB-ID): 1

Transmettre le total de contrôle

Confirmation

Lire — Requête à l'initialisation

Ecrire — Répondre aux requêtes d'initialisation

Transmettre chaque cycle — Intervalle: T#50ms

Transmettre au cas de modification — Intervalle minimum: T#20ms

Transmettre au cas d'événement — Variable:

Effacer liaison au réseau

OK — Annuler

<u>Liaison au fichier:</u>

Nom de fichier : Si vous disposez déjà d'un fichier d'exportation (*.exp) ou d'un fichier DCF contenant les variables souhaitées, vous pouvez les intégrer. Introduisez à cet effet le chemin d'accès correspondant ou aidez-vous de la boîte de dialogue standard via le bouton **Balayer**. Lors de la lecture, les fichiers DCF sont convertis en syntaxe CEI.

Choisissez l'option **Importer avant la compilation** si vous souhaitez que la liste externe de variables soit lue avant chaque compilation du projet. Choisissez l'option **Exporter avant la compilation** si vous souhaitez que la liste de variables soit à nouveau écrite dans le fichier externe indiqué avant chaque compilation du projet.

Lorsque vous refermez le dialogue 'Liste des variables globales' avec OK, le nouvel objet est créé dans l'Organisateur d'objets, muni du symbole -, et cet objet peut être ouvert au moyen de la commande 'Projet' 'Objet' 'Éditer' ou encore par un double-clic sur l'entrée correspondante.

La commande 'Projet' 'Objet' 'Caractéristiques' vous permet d'ouvrir à nouveau le dialogue de configuration 'Liste de variables globales' se rapportant à l'entrée marquée dans l'Organisateur d'objets.

Configuration de variables réseau :

ATTENTION ! Pour les variables réseau globales, il n'est actuellement pas possible d'effectuer des changements En ligne. Les changements dans la configuration des variables réseau ne sont pas reconnus comme des modifications du projet ! (Défaut Id 3320)

Si l'option 'Supporter variable de réseau' est activee dans la Configuration de Cible, le bouton **Ajouter Liaison au réseau** est disponible. Appuyer ce bouton extende la boite de dialogue, voir l'image ci-dessus. Si l'option n'est pas activee, le bouton manque.

Connection <n> (<Netzwerktyp>):

Dans la partie inférieure de la boîte de dialogue, un jeu de configurations peut être créé pour un total de quatre (n=1 à 4) connexion réseau sur quatre onglets, déterminant la façon dont la liste de variables doit être traitée dans l'échange avec les autres utilisateurs réseau. Pour que l'échange se passe comme convenu, cette liste de variables doit être configurée de façon adéquate auprès des autres utilisateurs réseau.

S'il n'y a pas encore de configuration, vous obtenez tout d'abord un seul onglet portant l'inscription **'Connexion 1 (UDP)'** dans le cas d'un réseau basé sur le protocole UDP. À chaque pression sur le bouton 'Ajouter connexion réseau', vous obtenez jusqu'à quatre onglets supplémentaires portant l'inscription „Connexion" suivie d'un numéro courant.

Type de réseau : Vous pouvez sélectionner le type souhaité hors de la liste. Cette liste est fonction de la configuration du système cible. Vous pouvez choisir par exemple "CAN" comme abréviation pour réseau CAN ou encore "UDP" pour un réseau avec protocole UDP.

Environnement : ce bouton donne accès à la boîte de dialogue **Environnement pour <type de réseau>** disposant des possibilités de configuration suivantes :

UDP:

Utiliser le standard : Si vous appuyez sur ce bouton, le port 1202 sera inscrit comme étant le port pour l'échange avec tous les autres utilisateurs réseau. L'adresse de diffusion / diffusion sélective par défaut est "255.255.255.255", ce qui signifie que l'échange se produit avec tous les utilisateurs réseau.

Port : Il s'agit d'une alternative au port par défaut (voir ci-dessus) (Attention : il doit être réglé de la même manière auprès de tous les nœuds concernés). Pour les réseaux UDP, une valeur introduite ici est reprise automatiquement pour toutes les connexions qui seront le cas échéant définies sur d'autres onglets.

Adresse Broadcast/Multicast Il s'agit d'une alternative à l'adresse par défaut (voir ci-dessus) ou zone d'adresse de sous-réseau (p.ex. "197.200.100.255" engloberait tous les utilisateurs avec une adresse IP 197.200.100.x.).

Notez que pour les systèmes Win32, l'adresse de diffusion / diffusion sélective vers le masque de sous-réseau de la configuration TCP/IP du PC doit correspondre !

Dialogue 'Environnement pour UDP'

Les options suivantes peuvent être (dés)activées pour la configuration de la transmission des variables.

Comprimer les variables : Les variables sont regroupées en paquets (télégrammes) pour la transmission, la taille de ces paquets étant dépendante du réseau. Si cette option est désactivée, un paquet est crée pour chaque variable.

Identificateur de la liste de variables : Numéro d'identification du premier paquet dans lequel des variables sont transmises. (valeur par défaut = 1). Les paquets suivants sont numérotés en séquence.

Transmettre le total de contrôle : Lors de l'envoi, chaque paquet est accompagné d'un total de contrôle qui sera vérifié à la réception. On détermine ainsi si les définitions de variables sont identiques chez l'émetteur et chez le récepteur. Les paquets avec un total de contrôle erroné ne seront pas repris et recevront un accusé de réception négatif pour autant que cette option soit choisie (voir ci-dessous, Confirmation de transfert).

Confirmation : Chaque message est confirmé par un accusé de réception du récepteur. Si l'émetteur n'a pas reçu un accusé de réception au minimum au sein d'un cycle, un message d'erreur et émis.

Lire : Les variables de la liste sont lues ; si l'option est désactivée, les nouvelles valeurs de variables envoyées par le biais du réseau sont ignorées.

Requête à l'initialisation : Si le noeud local peut lire (option Lire activée, voir ci-dessus) et est redémarré, les valeurs actuelles de variables sont demandées (de manière optionnelle) aux autres automates pouvant écrire ces mêmes valeurs, et envoyées à partir de ces automates, indépendamment des autres conditions temporelles ou événementielles sous lesquelles elles seraient normalement envoyées conformément à la configuration. Il faut pour ce faire que l'option de réponse à une demande d'initialisation soit activée dans la configuration des variables des automates pouvant écrire les valeurs (voir ci-dessous).

Ecrire : Les variables de la liste sont écrites, les options suivantes étant disponibles :

Répondre aux requêtes d'initialisation : Si le noeud local peut écrire (option Écrire activée, voir ci-dessus), il peut être donné une réponse aux requêtes provenant de noeuds pouvant lire et envoyées par ces derniers dans le cadre d'une initialisation (option de Requête à l'initialisation, voir ci-dessus). Ce qui signifie que les valeurs actuelles de variables sont transmises, même si les autres conditions temporelles ou événementielles telles que configurées ne l'exigent pas.

Transmettre chaque cycle : Les variables sont écrites conformément aux intervalles donnés à la rubrique Intervalle (exemple T#70ms).

Transmettre en cas de modification : Les variables ne sont écrites que lorsqu'un changement a eu lieu, et on peut cependant déterminer un intervalle minimum de temps entre les transmissions dans le champ Intervalle.

Transmettre en cas d'événement : Les variables de la liste sont écrites si la variable introduite sous **Variable** a la valeur TRUE.

Lorsque vous refermez la boîte de dialogue 'Liste de variables globales' avec **OK**, le nouvel objet est créé. Vous reconnaissez les listes de variables réseau globales au symbole dans l'Organisateur d'objets. À l'aide de la commande 'Projet' 'Objet' 'Propriétés', vous pouvez ouvrir à nouveau la boîte de dialogue relative à l'entrée marquée dans l'Organisateur d'objets.

Remarque : Si une variable réseau globale est utilisée dans une ou plusieurs **tâches**, la composante temporelle de la transmission est soumise aux conditions suivantes : Lors de l'appel de chaque tâche, on vérifie quels paramètres sont d'application pour la transmission de la valeur de la variable (configuration par le biais de la boîte de dialogue 'Liste de variables globales'). La valeur de variable est transmise ou non en fonction de l'intervalle écoulé à l'instant. Lors de chaque transmission, le compteur pour les intervalles de cette variable est remis à zéro.

L'envoi s'effectue toujours au départ du système d'exécution de l'automate concerné. Des fonctions spécifiques aux automates ne doivent dès lors pas être disponibles pour l'échange de données.

ATTENTION ! Pour les variables réseau globales, il n'est actuellement pas possible d'effectuer des changements En ligne. Les changements dans la configuration des variables réseau ne sont pas reconnus comme des modifications du projet ! (Défaut Id 3320)

Édition des listes de variables globales, variables réseau globales

L'éditeur pour les variables globales travaille de la même façon que l'éditeur de déclaration. Dans le cas de l'affichage d'une liste de variables externe, vous ne pouvez pas l'éditer à cet endroit. Les listes de variables externes peuvent uniquement être traitées de manière externe et sont lues de nouveau lors de chaque ouverture et de chaque compilation du projet.

Syntaxe:

VAR_GLOBAL
(* Déclarations de variables *)
END_VAR

Les variables réseau ne peuvent être utilisées que si le système cible le permet, et elles sont également définies à l'aide de cette syntaxe.

Exemple de liste de variable réseau créée par l'insertion d'un fichier d'exportation *.exp, et ayant reçu le titre NETWORKVARIABLES_UDP :

Exemple d'une liste de variables réseau

Network_Vars_UDP D:\NetworkGlobalVars UDP.EXP

```
VAR_GLOBAL CONSTANT
    MAX_NetVarItems_UDP   : INT := 0;
    MAX_NetVarPDO_Rx_UDP    : INT := 0;
    MAX_NetVarPDO_Tx_UDP    : INT := 0;
    MAX_NetVarOD_UDP      : INT := 0;
END_VAR
VAR_GLOBAL
    pNetVarItems_UDP      : ARRAY[0..MAX_NetVarItems_UDP] OF NetVarDataItem_UDP;
    pNetVarPDO_Rx_UDP       : ARRAY[0..MAX_NetVarPDO_Rx_UDP] OF NetVarPDO_Rx_UDP;
    pNetVarPDO_Tx_UDP       : ARRAY[0..MAX_NetVarPDO_Tx_UDP] OF NetVarPDO_Tx_UDP;
    pNetVarOD_UDP         : ARRAY[0..MAX_NetVarOD_UDP] OF NetVarSDO_UDP;
END_VAR
```

Édition des listes de variables globales rémanentes

Si le système d'exécution le permet, vous pouvez travailler avec des variables rémanentes. Il existe deux sortes de variables rémanentes:

- Les **Variables rémanentes** sont gardées même après une interruption non voulue du système d'exécution (on/off) ou une commande 'En Ligne' 'Reset' dans CoDeSys.
- Les **Variables persistantes** sont gardées après un téléchargement.

Le mot-clé RETAIN ou PERSISTENT est ajouté aux **variables rémanentes.**

Attention: Les variables persistantes ne sont pas nécessairement des variables rémanentes !

Syntaxe:

VAR_GLOBAL RETAIN
(* Déclarations de variables *)
END_VAR

VAR_GLOBAL PERSISTENT
(* Déclarations de variables *)
END_VAR

Les variables réseau (spécifiques au système cible) sont également définies à l'aide de cette syntaxe.

Constantes globales

Le mot clé **CONSTANT** est ajouté aux constantes globales.

Syntaxe:

VAR_GLOBAL CONSTANT
(* Déclarations de variables *)
END_VAR

6.2.2 Variable Configuration

Dans les blocs fonctionnels, il est possible de spécifier des adresses d'entrée et de sortie définies de manière incomplète, entre les mots clés **VAR** et **END_VAR**. Les adresses qui sont définies de manière incomplète sont caractérisées par une étoile.

Exemple :

```
FUNCTION_BLOCK locio
VAR
 loci AT %I*: BOOL := TRUE;
 loco AT %Q*: BOOL;
END_VAR
```

Dans le cas présent, deux variables locales d'entrée-sortie sont définies, une variable local-In (%I*) et une variable local-Out (%Q*).

Au cas où vous souhaitez configurer des variables locales d'entrée-sortie, vous disposez, à l'état standard, de l'objet **Variable_Configuration** dans l'onglet **Ressources** de l'Organisateur d'objets. Le nom de l'objet peut être modifié et des objets supplémentaires pour configuration de variables peuvent être crées.

L'éditeur pour la configuration des variables travaille de la même façon que l'éditeur de déclaration.

Les variables nécessaires à la configuration de variables locales d'entrée-sortie doivent figurer entre les mots clés **VAR_CONFIG** et **END_VAR**.

Le nom d'une telle variable est composé d'un chemin d'instance complet et les différents noms de module et d'instance sont séparés par des points. La déclaration doit contenir une adresse, dont la classe (Entrée/Sortie) correspond à l'adresse spécifiée de manière incomplète (%I*, %Q*) dans le bloc fonctionnel. Le type de données doit lui aussi correspondre à la déclaration au niveau du bloc fonctionnel.

Les variables de configuration dont le chemin d'instance n'est pas valide, du fait que l'instance n'existe pas, sont considérées comme des erreurs et sont caractérisées comme telles. Une erreur est également signalée dans le cas inverse, à savoir lorsque aucune configuration n'est attribuée à une variable d'instance. Pour obtenir une liste complète de toutes les variables de configuration requises, vous pouvez utiliser la commande 'Tous les chemins d'instance' dans le menu 'Insérer'.

Exemple de configuration de variable

Soit un programme avec les définitions de blocs fonctionnels suivantes:

```
PROGRAM PLC_PRG
VAR
 Hugo: locio;
 Otto: locio;
END_VAR
```

Une configuration des variables correcte ressemblerait alors à ceci:

```
VAR_CONFIG
 PLC_PRG.Hugo.loci AT %IX1.0 : BOOL;
 PLC_PRG.Hugo.loco AT %QX0.0 : BOOL;
 PLC_PRG.Otto.loci AT %IX2.0 : BOOL;
```

```
  PLC_PRG.Otto.loco AT %QX0.3 : BOOL;
END_VAR
```

Remarque : Il faut veiller à ce que les sorties utilisées dans la configuration des variables ne soient pas définies directement dans le projet ou par l'intermédiaire de variables (déclaration AT) car ces définitions seront négligées.

'Insérer' 'Tous les chemins d'instance'

Cette commande permet de créer un bloc **VAR_CONFIG – END_VAR**, qui contient tous les chemins d'instance présents à l'intérieur du projet. Les déclarations existantes ne sont pas insérées à nouveau, afin de conserver les adresses existantes. Cette option est disponible dans la fenêtre de la configuration des variables, lorsque le projet a été compilé ('Projet' 'Compiler tout').

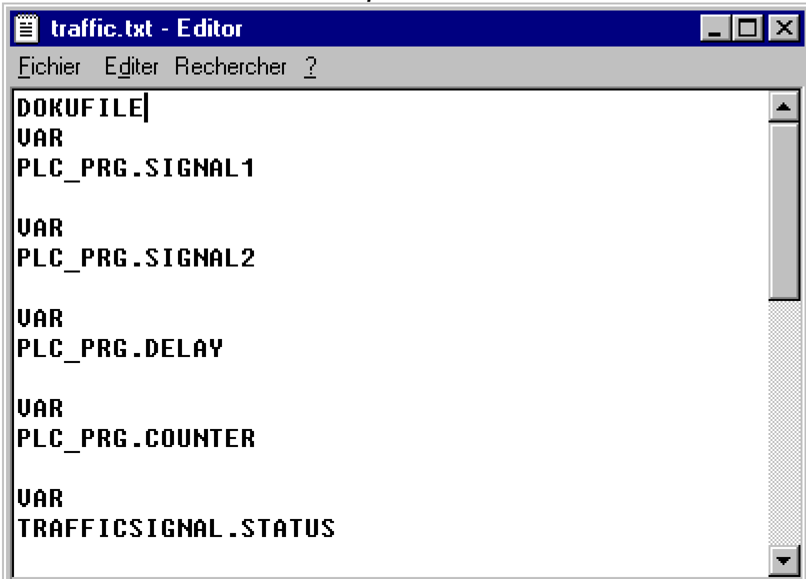
6.2.3 Fichier cadre pour la documentation

Si vous devez documenter un projet à plusieurs reprises, vous pouvez, outre la fonction 'Projet' 'Traduire dans une autre langue', utiliser le document modèle. Il se peut également que vous ayez besoin de la documentation pour le même projet avec des commentaires se rapportant aux variables dans plusieurs langues, ou encore vous souhaitez documenter plusieurs projets similaires qui utilisent les mêmes noms de variables.

Sélectionnez la commande 'Extras' 'Etablir fichier cadre pour la documentation'.

Le fichier ainsi constitué peut être chargé et édité dans l'éditeur littéral de votre choix. Le fichier débute par la ligne **DOKUFILE**, qui est suivie par une liste des variables du projets. Pour chacune de ces variables, trois lignes sont prédéfinies: une ligne **VAR**, qui indique le début d'une nouvelle variable, ensuite une ligne avec le nom de la variable, et enfin une ligne vide. Vous pouvez à présent remplacer cette ligne par un commentaire se rapportant à la variable. Les variables que vous ne voulez pas voir documentées peuvent tout simplement être enlevées du texte. Vous avez la possibilité de créer autant de fichiers cadres pour la documentation que vous le souhaitez.

Editeur Windows avec fichier cadre pour la documentation



Pour utiliser un fichier cadre pour la documentation, effectuez la commande 'Extras' 'Sélectionner fichier cadre pour la documentation'. Si vous voulez à présent documenter la totalité de votre projet, ou imprimer des parties de votre projet, alors le commentaire que vous avez créé au niveau du fichier cadre pour la documentation de cette variable est inséré dans la partie implémentation (pas la partie déclaration !), à l'endroit même où cette variable est utilisée. Ce commentaire n'apparaît qu'à l'impression!

'Extras' 'Etablir fichier cadre pour la documentation'

Cette commande vous permet de créer un fichier cadre pour la documentation. Cette commande est disponible lorsqu'un objet a été sélectionné dans Variables globales.

La boîte de dialogue pour l'enregistrement des fichiers sous un nouveau nom s'affiche. Dans le champ **Nom de fichier**, l'extension *.txt a déjà été entrée. Choisissez un nom à votre convenance. Un fichier texte est créé, dans lequel sont répertoriées toutes les variables de votre projet.

'Extras' 'Sélectionner fichier cadre pour la documentation'

Cette commande vous permet de sélectionner un fichier cadre pour la documentation.

La boîte de dialogue pour l'ouverture des fichiers est affichée. Sélectionnez le fichier cadre pour la documentation et appuyez sur **OK**. Si vous voulez à présent documenter la totalité de votre projet, ou imprimer des parties de votre projet, alors le commentaire que vous avez créé au niveau du fichier cadre pour la documentation est inséré dans le texte de programmation, pour chacune des variables. Ce commentaire n'apparaît qu'à l'impression!

Pour créer un fichier cadre pour la documentation, vous pouvez utiliser la commande 'Extras' 'Etablir fichier cadre pour la documentation'.

6.3 Configuration de l'alarme

6.3.1 Aperçu de la configuration de l'alarme

À l'aide du système d'alarme intégré à CoDeSys, il est possible de détecter des états de processus critiques, d'enregistrer ceux-ci ou de les illustrer dans une visualisation pour l'utilisateur. Le traitement de l'alarme peut être effectué dans CoDeSys mais également sur l'automate programmable, cette dernière possibilité étant une option. Pour le traitement de l'alarme sur l'automate programmable, reportez-vous à la configuration du système cible dans le dialogue 'Visualisation'.

Pour la configuration, vous disposez de la 'Configuration d'alarme' dans l'Organisateur d'objets, dans le registre Ressources.

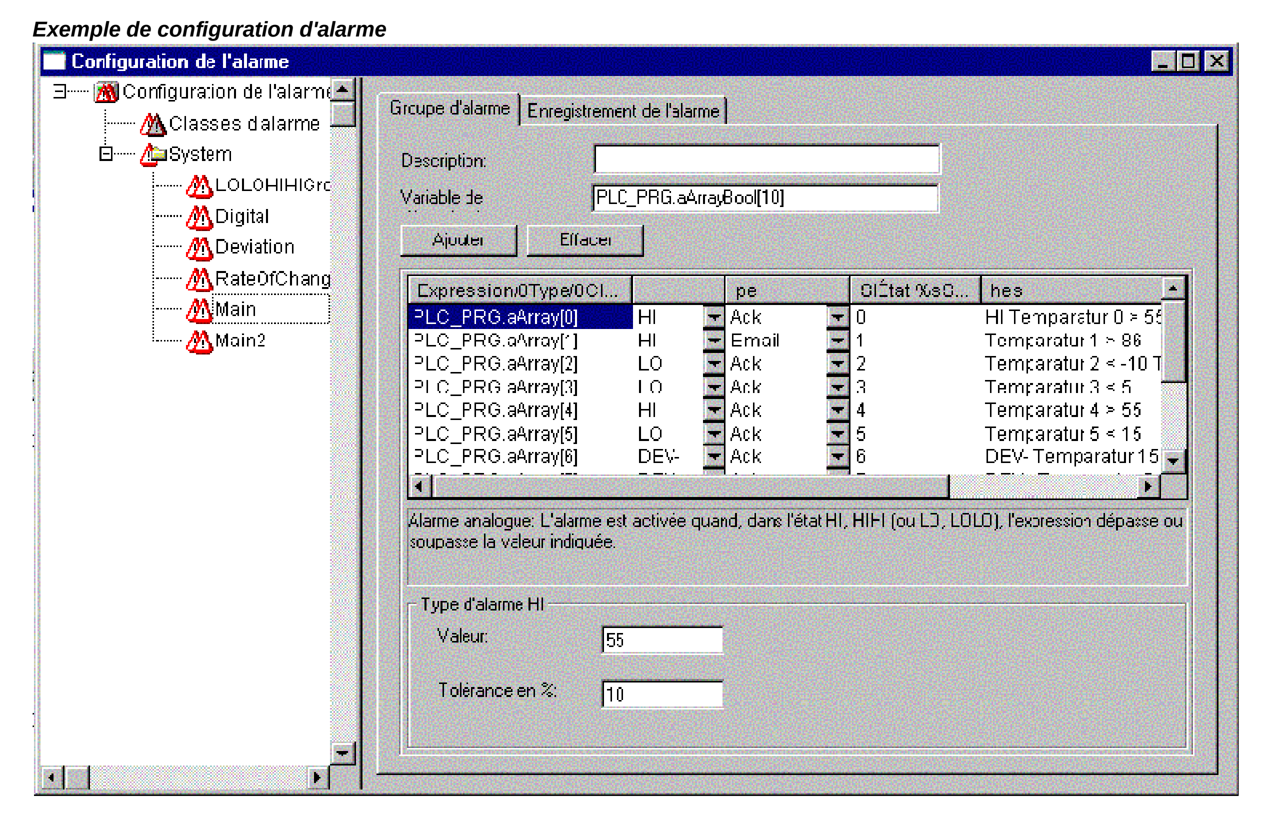
C'est ici que les **Classes d'alarme** et les **Groupes d'alarme** sont définis. La classe d'alarme sert à la standardisation d'une alarme, elle confère en d'autres termes des paramètres précis à cette alarme. Le groupe d'alarme sert à la configuration concrète d'une ou de plusieurs alarmes (auxquelles une classe précise et d'autres paramètres sont attribués) afin qu'elles soient utilisables au sein du projet. Il offre ainsi la possibilité de structurer les alarmes disponibles. Les différents groupes d'alarme sont rattachés et définis par l'utilisateur sous le point '**Système**' de la structure.

L'élément de 'Tableau d'alarme' sert à **visualiser** les alarmes. Ce tableau permet à l'utilisateur de surveiller et de confirmer les alarmes.

Pour obtenir un **historique** ou un enregistrement des événements d'alarme dans un fichier journal, un tel fichier doit être signalé, et le Comportement à la sauvegarde doit être défini pour chaque groupe d'alarme.

Si vous sélectionnez l'entrée 'Configuration de l'alarme' au sein des Ressources, vous avez accès au **dialogue 'Configuration de l'alarme'** contenant une fenêtre scindée en deux, et dont le mode de fonctionnement correspond à celui de la configuration de l'automate ou à celui de la configuration des tâches. L'arborescence de configuration est donnée à gauche, tandis que la partie droite reprend le dialogue de configuration correspondant à l'entrée sélectionnée dans l'arborescence.

Exemple de configuration d'alarme



Ouvrez l'arborescence actuellement disponible en cliquant sur le symbole Plus en regard de l'entrée 'Configuration de l'alarme'. Dans le cas d'une nouvelle configuration, cette arborescence ne contient que les entrées 'Classes d'alarme' et 'Système'.

6.3.2 Système d'alarme

L'utilisation d'un système d'alarme dans CoDeSys répond aux descriptions et définitions générales relatives aux alarmes:

- **Alarme**: une alarme est généralement considérée comme étant une condition spéciale (valeur d'expression).
- **Priorité**: la priorité ou "Severity" d'une alarme décrit à quel point la condition d'alarme est importante ou lourde de conséquences. La plus grande priorité est "0", la moins urgente est "255".
- **État d'alarme** : une expression / variable configurée pour la surveillance par alarme peut prendre les états suivants: NORM (pas d'état d'alarme), INTO (une alarme est intervenue, "l'alarme entre"), ACK (l'alarme est intervenue et a été confirmée par l'utilisateur), OUTOF (l'état d'alarme est terminé, "l'alarme est sortie", mais pas encore confirmée!)
- **Sous-état** : une condition d'alarme peut disposer de limites (Lo, Hi) et de limites "extrêmes" (LoLo, HiHi). Exemple : la valeur d'une expression augmente et dépasse tout d'abord la limite Hi, ce qui déclenche l'alarme Hi. Si la valeur continue à augmenter et dépasse la limite HiHi avant même que l'alarme Hi n'ait pu être confirmée, cette dernière se confirme automatiquement en interne et il ne reste plus que l'état d'alarme HiHi dans la liste des alarmes (liste interne de gestion des alarmes). L'état Hi est désigné dans ce cas par les termes de sous-état
- **Confirmation d'alarmes** : une application principale de l'alarme est de communiquer une situation d'alarme à l'utilisateur. À cet égard, il s'avère souvent nécessaire de s'assurer que l'information à bien été reçue (voir actions possibles dans la configuration des classes d'alarme). L'utilisateur doit "confirmer" l'alarme afin que cette dernière puisse être effacée de la liste des alarmes.
- **Événement d'alarme** : un événement d'alarme ne doit pas être confondu avec une condition d'alarme. Alors qu'une condition d'alarme peut s'étendre sur une durée plus longue, un événement d'alarme décrit l'apparition momentanée d'une modification, par exemple le passage de l'état normal à l'état d'alarme. Événements d'alarme possibles: INTO, OUTOF, ACK,

Les possibilités ci-dessous sont supportées par CoDeSys.

- Désactivation de la création d'alarme d'une seule alarme ou de groupes d'alarme entiers
- Sélection des alarmes à représenter par le biais des groupes d'alarme, ainsi que de la priorité d'alarmes isolées
- Enregistrement de tous les Événements d'alarme survenus
- Élément de visualisation Tableau d'alarme dans la visualisation CoDeSys

6.3.3 Classes d'alarme

Les classes d'alarmes servent à la description générale de certains critères d'alarme comme par exemple la philosophie de l'acquittement (confirmation d'une alarme par l'utilisateur), l'exécution de l'action (ce qui doit se passer automatiquement avec certains états d'alarme) et la visualisation au sein du Tableau d'alarme. Les classes d'alarmes sont définies de manière globale dans la Configuration de l'alarme et font alors office de "Configuration de base" pour chaque groupe d'alarme.

Configuration des classes d'alarmes :

marquez l'entrée 'Classes d'alarmes' dans l'arborescence de configuration d'alarmes. Cela vous donne accès au dialogue de configuration des 'Classes d'alarmes':

Dialogue de configuration des 'Classes d'alarmes'

Configuration de l'alarme

Configuration de
Classes d'
System
LOLO
Digita
Devia
RateC
Main
Main2

Classes d'alarme

Ajouter Effacer

Nom	Accusé de réception
DEFAULT	ACK_INTO
FatalError	NO_ACK
Warning	ACK_INTO
Email	ACK_INTO
Call	NO_ACK
Ack	ACK_INTO

Actions assignées pour classe DEFAULT

Actions possibles:
Enregistrer
imprimer
message
courrier électronique
variable
exécuter

> >> < << ...

Actions assignées:

Evénéments pour action: INTO ACK OUTOF

Couleurs/Bitmaps pour classe DEFAULT

INTO ACK OUTOF
Couleur de front:
Couleur de fond:
Bitmap:
Supprimer bitmaps

Appuyez sur le bouton **Ajouter** pour créer une classe d'alarme. Une ligne est alors ajoutée dans la fenêtre supérieure, n'affichant tout d'abord que le paramétrage "NOACK" (no acknowledgement = pas d'acquittement) dans la colonne 'Acquittement'. Attribuez un nom à la classe d'alarme ; cliquez pour ce faire sur le champ sous **Nom**, ce qui vous donne accès à un cadre d'édition, choisissez si nécessaire un autre type d'acquittement dans la liste de sélection proposée sous **Accusé de réception**.

Les types d'acquittement suivants sont possibles :

NO_ACK: une confirmation de l'alarme par l'utilisateur n'est pas nécessaire

ACK_INTO: une "condition d'alarme entrant" (statut "INTO", l'alarme commence) doit être acquittée par l'utilisateur.

ACK_OUTOF: une "condition d'alarme sortant" (statut "OUTOF", l'alarme se termine) doit être acquittée par l'utilisateur.

ACK_ALL: toutes les conditions d'alarme (entrant ou sortant) doivent être acquittées par l'utilisateur.

Vous pouvez en outre saisir un **commentaire**.

Les entrées relatives aux autres classes d'alarme sont chaque fois rattachées en bas de la liste.

Le bouton **Effacer** vous permet de supprimer l'entrée sélectionnée à l'instant.

Actions assignées pour la classe <Nom de la classe>:

Chaque classe d'alarme peut se voir attribuer une liste d'actions qui seront déclenchées dès que des événements d'alarmes interviennent.

Marquez dans la liste **Actions possibles** celle que vous souhaitez et transférez-la dans le champ **Actions assignées** par le biais du bouton "**>**". Le bouton "**>>**" vous permet de sélectionner simultanément toutes les actions. De même, vous pouvez supprimer par le biais de "**<**" ou de "**<<**" une ou encore toutes les actions de la sélection.

Lorsqu'une action sélectionnée est marquée dans la liste, le bouton "**...**" vous donne accès au dialogue correspondant afin d'y définir l'adresse e-mail souhaitée, la configuration de l'imprimante ainsi qu'un texte de message.

Les types d'action ci-après sont supportés :

Action	Description	Paramétrage à effectuer dans le dialogue correspondant :
Enregistrer:	L'événement d'alarme est enregistré en interne afin de pouvoir l'afficher par exemple dans un Fichier de journal. Veuillez noter : ce fichier doit pour ce faire être défini dans la Configuration des groupes d'alarme !	Ces paramétrages doivent être effectués lors de la configuration de l'Enregistrement de l'alarme.
Imprimer:	Un message est délivré au niveau de l'imprimante.	Interface de l'imprimante: sélectionnez une des imprimantes définies dans le système local; Texte de sortie: texte de message (voir plus bas) qui doit être imprimé
Message:	Une boîte de message s'affiche avec le texte à définir.	Message: texte de message (voir plus bas) qui doit être édité dans une propre fenêtre de messages.
E-Mail:	Un message électronique est envoyé, contenant le	De: adresse e-mail du destinateur ; À : adresse e-mail du destinataire ; Concerne : texte de référence; Message: texte du message (voir plus

	texte à définir.	bas); Serveur: nom du serveur e-mail
Variables:	Le statut d'alarme ou un texte de message est attribué à une variable du projet actuel.	Variables: Nom de variable: une variable peut être sélectionnée par le biais de la liste de sélection pour l'édition (<F2>) : Une variable booléenne peut être utilisée pour afficher les statuts d'alarme NORM=0 et INTO=1, une variable entière indique les statuts d'alarme NORM =0, INTO =1, ACK =2, OUTOF =4; une variable de type string est attribuée au texte de message défini dans le champ Message (voir plus bas)
Exécuter :	un programme externe est démarré dès qu'un événement d'alarme (voir plus bas) intervient.	Fichier exécutable : nom du fichier qui doit être exécuté (p.ex. notepad.exe); le bouton "..." permet d'appeler à l'aide le dialogue standard de sélection de fichier; Paramètres: les paramètres correspondant au fichier exe qui doivent être rattachés à l'appel.

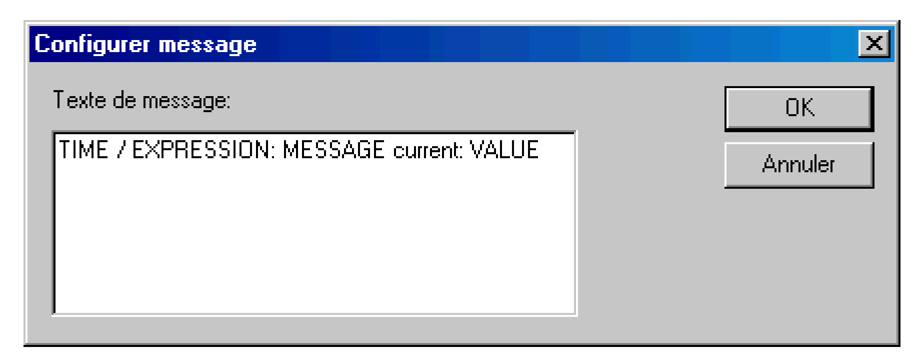
Définition des Textes de message:

Lors de la configuration des actions 'Message','E-mail', 'Imprimer', 'Variable' et éventuellement 'Exécuter', vous pouvez définir un texte de message qui sera affiché dès qu'un événement d'alarme (voir plus bas) intervient. Il faut insérer des retours à la ligne grâce à <Ctrl>+<Enter>. Les **espaces réservés** ci-dessous peuvent être utilisés :

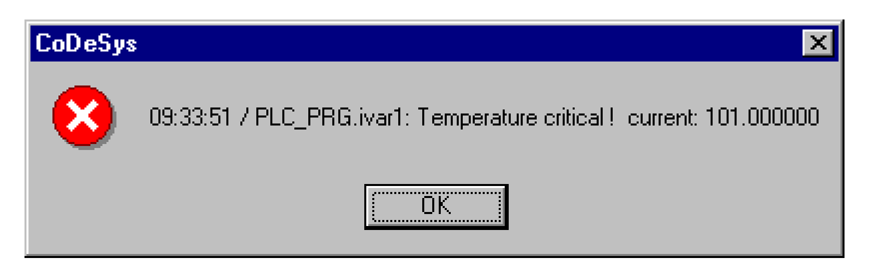
MESSAGE	Le texte de message défini pour l'alarme dans la Configuration des groupes d'alarme (colonne **Messages**) apparaît à l'écran.
DATE	La date du changement vers le statut correspondant (INTO)
TIME	L'heure effective du déclenchement de l'alarme apparaît à l'écran
EXPRESSION	L'expression (définie dans le groupe d'alarme) qui a déclenché l'alarme.
PRIORITY	Priorité de l'alarme (définie dans le groupe d'alarme)
VALUE	Valeur actuelle de l'expression
TYPE	Type d'alarme (défini dans le groupe d'alarme)
CLASS	Classe d'alarme (définie dans le groupe d'alarme)
TARGETVALUE	Valeur cible des types d'alarme DEV+ et DEV- (définie dans le groupe d'alarme)
DEADBAND	Tolérance de l'alarme (définie dans le groupe d'alarme)
ALLDEFAULT	Toutes les données relatives à l'alarme apparaissent à l'écran, comme décrit dans l'édition dans un Fichier de sauvegarde (Historique)

Exemple :

Saisissez les éléments suivants dans la fenêtre des messages, cela pour la définition de la boîte de message (action 'Message') :



Saisissez en outre ce qui suit pour la définition de l'alarme dans le groupe d'alarmes, dans le champ de tableau 'Message': "Temperature critical !". Le message d'alarme apparaîtra alors comme suit à l'écran :



Remarque: Le texte de message peut également être pris en compte lors de la commutation du projet à une autre langue, cela par le biais d'un fichier *.vis ou d'un fichier de traduction *.tlt. Afin d'être repris dans le fichier de traduction *.tlt, la chaîne de caractère correspondante doit cependant être pourvue du symbole "#" au début et à la fin, comme pour tous les textes se rapportant aux visualisations (p.ex. dans l'exemple ci-dessus: "#Temperature critical!#" et "TIME /EXPRESSION: MESSAGE #current#: VALUE", cela afin de conserver les parties adéquates de texte comme ALARMTEXT_ITEM dans le fichier de traduction.)

Un **fichier de sauvegarde** pour l'action 'Enregistrer' est défini au sein de la configuration des groupes d'alarme.

Événements pour les actions :

Pour chaque action, il faut déterminer les **événements d'alarme** déclencheurs.

Activez les événements souhaités :

INTO L'alarme commence.

ACK Il y a confirmation par l'utilisateur.

OUTOF L'état d'alarme est terminé.

Couleurs/Bitmaps pour classe <nom de la classe>

Chaque classe d'alarmes peut se voir attribuer différentes couleurs et différents bitmaps, permettant de faire une distinction entre les alarmes lors de la visualisation du Tableau d'alarme. Choisissez à chaque fois une **Couleur de front** et une **Couleur de fond** pour les événements possibles d'alarme INTO, ACK et OUTOF (voir plus haut). Si vous cliquez sur les flèches, vous avez accès au dialogue standard de sélection de couleur, et vous avez accès au dialogue de sélection des fichiers bitmap si vous cliquez sur le carré gris.

6.3.4 Groupes d'alarme

Les groupes d'alarmes servent à structurer les différentes alarmes. Chaque alarme est attribuée de manière stricte à un seul groupe d'alarme et est gérée par ce dernier. Toutes les alarmes au sein d'un groupe peuvent recevoir une seule et même variable de désactivation ainsi que des paramètres communs pour l'enregistrement de l'alarme. Le groupe peut donc servir à la structuration des alarmes disponibles. Même une alarme individuelle doit être configurée dans un groupe d'alarmes.

La structure hiérarchique des groupes d'alarmes dans la configuration d'alarme peut être construite à l'aide d'éléments de dossiers. Lorsqu'un groupe d'alarme est sélectionné dans l'arborescence des alarmes, le **dialogue de Groupe d'alarme** est automatiquement affiché :

Dialogue de configuration 'Groupe d'alarme'

Configuration de l'alarme

Configuration de l'al
Classes d'ala
System
LOLOHII
Digital
Deviatior
RateOfC
Main
Main2

Groupe d'alarme | Enregistrement de l'alarme

Description: Groupe1
Variable de désactivation: PLC_PRG.n
Ajouter Effacer

Expression	Type	Classe	Priorité	Message
PLC_PRG.rSinus	HI	Ack	0	Sinus > 8
PLC_PRG.rSinus	HIHI	Ack	0	
PLC_PRG.rSinus	LO	Ack	0	
PLC_PRG.rSinus	LOLO	Ack	0	

Alarme analogue: L'alarme est activée quand, dans l'état HI, HIHI (ou LO, LOLO), l'expression dépasse ou soupasse la valeur indiquée.

Type d'alarme HI
Valeur: 80
Tolérance en %:

Le champ **Description** vous permet de saisir une désignation pour le groupe d'alarme.

La **Variable de désactivation** peut être une variable booléenne de projet qui désactive avec un front montant le déclenchement pour toutes les alarmes du groupe présent, et qui l'active à nouveau avec un front descendant.

Le bouton **Ajouter** vous permet d'ajouter des alarmes individuelles au groupe, ces alarmes étant définies par les paramètres ci-dessous:

Expression: La variable de projet à laquelle l'alarme se rapporte. Aidez-vous de la liste de sélection pour l'édition <F2> ou de la fonction Intellisense pour une saisie correcte. Il est également possible de saisir une expression (p.ex. "a + b")

Type: Les types d'alarmes ci-dessous peuvent être utilisés : (Notez le commentaire correspondant affiché en bas du tableau)

DIG=0: Alarme numérique, devient active lorsque l'expression prend la valeur FALSE.

DIG=1: Alarme numérique, devient active lorsque l'expression prend la valeur TRUE.

LOLO: Alarme analogique, devient active lorsque l'expression dépasse par le bas la valeur indiquée sous 'type d'alarme LOLO'. Il est possible de saisir une tolérance (pourcentage) par rapport à cette valeur. Dans cette zone de tolérance, aucune alarme ne se déclenche, même si la valeur LOLO est dépassée par le bas.

LO: Identique à LOLO

HI: Alarme analogique, devient active lorsque l'expression dépasse la valeur indiquée sous 'type d'alarme HIHI'. Il est possible de saisir une tolérance (pourcentage) par rapport à cette valeur. Dans cette zone de tolérance, aucune alarme ne se déclenche, même si la valeur HI est dépassée.

HIHI: Identique à HI

DEV-: Déviation (déviation par rapport à la valeur cible); l'alarme devient active lorsque l'expression dépasse par le bas la valeur cible indiquée sous 'Type d'alarme DEV-' + une déviation exprimée en pourcentages. Pourcentage de déviation = valeur cible* (déviation en %) /100.

DEV+: Déviation (déviation par rapport à la valeur cible); l'alarme devient active lorsque l'expression dépasse la valeur cible indiquée sous 'Type d'alarme DEV-' + la déviation indiquée. Pourcentage de déviation = valeur cible* (déviation en %) /100.

ROC: Rate of Change (taux de changement par unité de temps); l'alarme devient active lorsque l'expression a trop fortement changé par rapport à la valeur précédente. La valeur limite de l'intensité du changement déclenchant l'alarme est définie par le nombre d'unités (changement de valeur) qui sont modifiées par seconde, minute ou heure.

Classe:Sélectionnez la Classe d'alarme souhaitée. Vous pouvez alors choisir entre les classes définies au sein de la configuration des classes d'alarme avant la dernière sauvegarde du projet.

Priorité: Vous pouvez ici saisir une priorité entre 0 et 255, 0 représentant la plus haute priorité. La priorité agit sur le tri au sein du tableau d'alarme.

Message:Définissez ici un texte pour le message qui pourra être affiché dans le tableau d'alarme ou par le biais de la macro "MESSAGE" au sein de chaque action. Cette boîte doit être confirmée par OK, ce qui ne confirme cependant pas automatiquement l'alarme ! Pour confirmer l'alarme, il faut avoir accès à la liste des alarmes, ce qui est possible via l'élément de visualisation 'Tableau d'alarme', ou encore via la date de l'entrée de l'alarme à reprendre dans un fichier de sauvegarde pouvant être créé en option.

Désactivation On peut saisir ici une variable de projet qui désactiverait le déclenchement de l'alarme avec un front montant. Notez cependant que cette entrée est écrasée par une saisie dans le champ 'Variable de désactivation' (voir plus haut).

6.3.5 Enregistrement de l'alarme

Un fichier peut être défini pour chaque groupe d'alarmes, ce fichier enregistrant les événements d'alarme pour autant (!) qu'un "Enregistrement" soit activé dans la Liste d'actions de la classe concernée.

Sélectionnez le groupe d'alarmes dans l'arborescence de configuration des alarmes puis choisissez l'onglet du dialogue 'Enregistrement de l'alarme':

Dialogue de configuration 'Enregistrement de l'alarme'

Les saisies suivantes sont possibles :

Chemin du fichier: Chemin du fichier indiqué sous le nom de fichier ; le bouton "**...**" vous donne accès au dialogue standard de sélection d'un répertoire. Si l'option spécifique à la cible 'Traitement de l'alarme au sein de l'automate programmable' est activée, le chemin donné sera ignoré et le fichier d'enregistrement sera créé dans le répertoire de téléchargement de l'automate programmable.

Nom du fichier: Nom du fichier dans lequel les événements d'alarme doivent être enregistrés (p.ex. "alarmlog"). Le fichier sera alors automatiquement créé avec le nom défini ici, et auquel on rattache un chiffre ainsi que l'extension ".alm". Le chiffre donne la version du fichier de journal. Le premier fichier d'enregistrement est suivi d'un 0, les autres, générés en raison des conditions décrites sous 'Événement de changement de fichier', sont suivis par ordre croissant de 1, 2, etc. (exemples -> "alarmlog0.alm", "alarmlog1.alm). Le format du fichier d'enregistrement peut être défini par le biais du dialogue de 'Configuration de cadre de documentation'.

Evénement de changement de fichier: Saisissez ici les conditions sous lesquelles un nouveau fichier d'enregistrement doit être créé. Conditions possibles : jamais, après une heure, après un jour, après une semaine, après un mois, après un front montant de la variable indiquée sous 'Variable déclencheur', après avoir atteint un nombre maximal d'entrées saisi sous 'Nombre maximal d'entrées'.

Variable de déclencheur ou **Nombre maximal d'entrées**: voir Événement de changement de fichier :

Effacer les données anciennes après .. Heures: nombre de jours suivant la date de la création, après lesquels tous les fichiers d'enregistrement d'alarme sont effacés, à l'exception du fichier en cours.

Le fichier d'enregistrement (Historique) contient les entrées suivantes.

Date/Heure en DWORD	Date	Heure	Événement	Expression	Type d'alarme	Valeur limite	Tolérance	Valeur act.	Classe	Priorité	Message
1046963332	6.3.03	16:08:52	INTO	PLC_PRG.b	LO	-30	5	-31	Alarm_high	0	Mess1
1046963333	6.3.03	16:08:53	ACK	PLC_PRG.n	HIHI	35			Avertissement	9	Mess2

Exemple :

```
1046963332,6.3.03 16:08:52,INTO,PLC_PRG.ivar5,HIHI,,,,    9.00,a_class2,0,
1046963333,6.3.03 16:08:53,INTO,PLC_PRG.ivar4,ROC,2,,,    6.00,a_class2,2,
1046963333,6.3.03 16:08:53,INTO,PLC_PRG.ivar3,DEV-,,,,   -6.00,a_class2,5,
1046963334,6.3.03 16:08:54,INTO,PLC_PRG.ivar2,LOLO,-35,,3,  -47.00,warning,10,warning: low
temperature !
1046963334,6.3.03 16:08:54,INTO,PLC_PRG.ivar1,HI,20,,5,   47.00,a_class1,2,temperature to
high ! Acknowledge !
```

6.3.6 Menu Extras: Configuration

<u>Catégorie Date/Temps:</u>

Vous pouvez définir ici le format dans lequel la date et l'heure doivent être affichés dans le Fichier d'enregistrement pour les alarmes. Saisissez les formats conformément à la syntaxe suivante, les traits d'union et deux-points sont enfermés entre des guillemets simples :

pour la date : dd'-'MM'-'yyyy -> p.ex. "12-Jan-1993"

pour l'heure : hh':'mm':'ss -> p.ex. "11:10:34"

<u>Catégorie Langage:</u>

Sélectionnez ici un fichier de langue qui doit être utilisé pour la commutation vers une autre langue et qui doit donc également contenir les textes de la configuration d'alarme. Reportez-vous pour ce faire aux descriptions ci-dessous :

- Visualisation, Configuration de la langue

- Traduire projet dans un autre langage

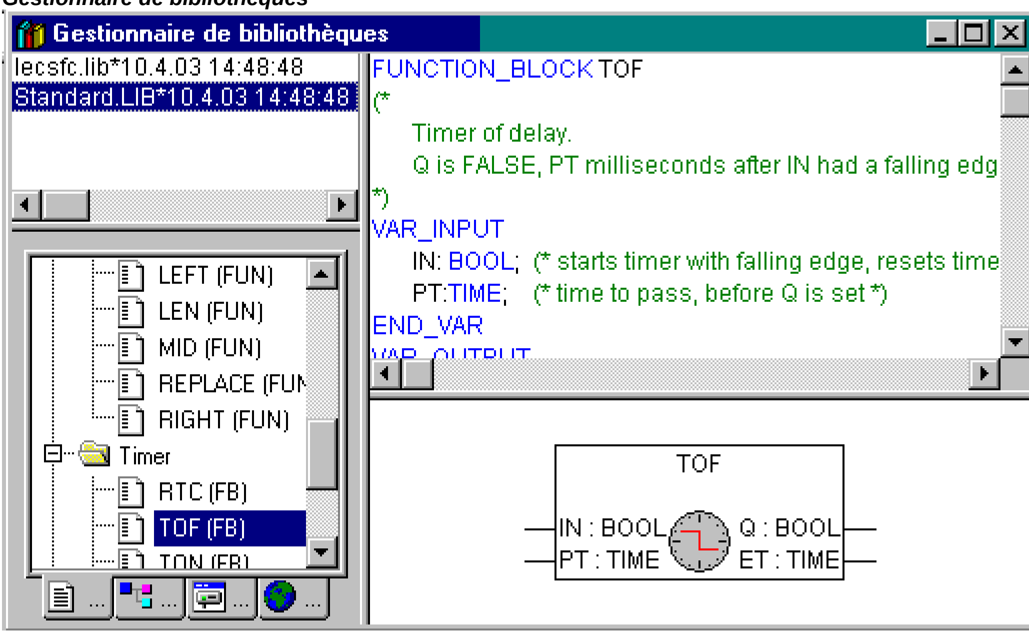
6.4 Gestionnaire de bibliothèques

Le gestionnaire de bibliothèques indique toutes les bibliothèques intégrées au projet actuel. Les modules, types de données et variables globales des bibliothèques peuvent être utilisés au même titre que des modules, types de données et variables globales définis directement dans le projet.

On accède au gestionnaire de bibliothèques au moyen de la commande '**Fenêtre**' '**Gestion des bibliothèques**' ou en sélectionnant l'onglet 'Ressources'.

Les informations relatives aux bibliothèques liées sont sauvegardées avec le projet et peuvent être consultées par le biais de la commande 'Extras' 'Propriétés' pour autant que l'entrée correspondante soit marquée dans le gestionnaire des bibliothèques.

Gestionnaire de bibliothèques



Utiliser le gestionnaire de bibliothèques

La fenêtre du gestionnaire de bibliothèques est scindée en trois ou quatre zones selon le cas, au moyen de barres de fractionnement. Dans la zone supérieure gauche sont répertoriées les bibliothèques intégrées au projet.

Dans la zone située immédiatement en dessous sont répertoriés les modules, types de données variables globales ou visualizations de la bibliothèque sélectionnée dans la zone du dessus, en fonction de l'onglet sélectionné.

Les dossiers sont ouverts ou fermés en double-cliquant sur la ligne correspondante ou en appuyant sur la touche <Entrée>. Devant chaque dossier fermé se trouve un signe plus, devant chaque dossier ouvert se trouve un signe moins.

Si un module est sélectionné en cliquant avec la souris ou au moyen de touches directionnelles, alors la déclaration du module est affichée dans la zone supérieure droite du gestionnaire de bibliothèques et la visualisation apparaît dans la zone inférieure, sous forme de graphique "black box" comprenant des entrées et des sorties.

Dans le cas des onglets Types de données et Variables globales, la déclaration est affichée dans la partie droite du gestionnaire de bibliothèques.

Bibliothèque standard

La bibliothèque 'standard.lib' est mise à votre disposition par défaut. La bibliothèque standard contient l'ensemble des fonctions et blocs fonctionnels requis par la norme CEI61131-3, dans laquelle sont définis les modules standard nécessaires à un système de programmation CEI. La différence entre une fonction standard et un opérateur réside dans le fait que l'opérateur est connu de façon implicite

par le système de programmation, alors que les modules standard doivent être intégrés au projet sous forme de bibliothèque (standard.lib).

Le code relatif à ces modules est présent sous forme de bibliothèque "C" et fait partie intégrante de CoDeSys.

Les bibliothèques définies par l'utilisateur

Un projet peut être enregistré en tant que bibliothèque par le biais de la commande '**Enregistrer sous'** dans le menu **'Fichier'.** Le projet lui-même reste inchangé, un fichier avec l'extension standard ".lib" est créé, qui sera par la suite mis à disposition sous le nom que vous lui aurez attribué, comme c'est le cas par exemple pour la bibliothèque standard.

Pour pouvoir utiliser les modules d'un projet dans le cadre d'un autre projet, elle est sauvegardée comme **Bibliothèque interne *.lib**. Elle peut alors être liée à un autre projet par le biais du gestionnaire de bibliothèques, comme c'est par exemple le cas pour la bibliothèque standard.lib.

Cependant, si vous avez implémenté des modules dans d'autres langages de programmation, par exemple C, et que vous souhaitez les relier à un autre projet par l'intermédiaire d'une bibliothèque, vous devez sélectionner lors de l'enregistrement du projet en question le type de fichier **Bibliothèque externe *.lib.** Cela signifie qu'un fichier supplémentaire en plus du fichier de bibliothèque sera créé portant également le nom de fichier de bibliothèque mais avec l'extension "*.h". Ce fichier est structuré comme fichier du langage de programmation C (Header-file) et comporte les déclarations de tous les modules, types de données et variables globales disponibles dans la bibliothèque. Si une bibliothèque externe est utilisée dans un projet, l'implémentation écrite pour les modules dans **CoDeSys** sera exécutée en mode simulation. Par contre, un système cible utilise l'implémentation écrite en langage C lors de l'exécution.

Si vous souhaitez soumettre une bibliothèque à une licence, vous devez appuyer sur le bouton **Edit info de licence...** et saisir les données adéquates dans la boîte de dialogue 'Éditer les informations relatives à l'attribution d'une licence'. Reportez-vous pour ce faire à la description de la commande 'Fichier' 'Enregistrer sous...' ou à la rubrique Gestion des licences dans CoDeSys.

'Insérer' 'Autre bibliothèque'

Cette commande permet d'associer une nouvelle bibliothèque à votre projet.

Dans la boîte de dialogue d'ouverture des fichiers, choisissez la bibliothèque souhaitée possédant l'extension "*.lib". Cette bibliothèque est désormais reprise dans la liste du gestionnaire de bibliothèques et vous pouvez utiliser les objets appartenant à cette bibliothèque au même titre que des objets définis directement dans le projet.

Si vous insérez une bibliothèque sujette à une licence, un message vous indique que cette bibliothèque n'est disponible qu'en mode démonstration ou qu'elle n'est pas valable pour le système cible tel qu'actuellement paramétré. Vous pouvez ignorer ce message ou prendre les mesures nécessaires quant à la licence. Des licences non valables créent un message d'erreur lors de la compilation du projet ('Projet' 'Compiler'). En double-cliquant sur le message d'erreur ou en appuyant sur <F4>, vous avez accès à la boîte de dialogue 'Informations relatives à l'attribution d'une licence' grâce à laquelle vous pouvez prendre les mesures nécessaires quant à cette licence.

Enlever une bibliothèque

La commande 'Editer' 'Supprimer' vous permet d'enlever une bibliothèque d'un projet et du gestionnaire de bibliothèques.

'Extras' 'Propriétés'

Cette commande donne accès à la boîte de dialogue 'Informations relatives à l'attribution d'une licence'. Dans le cas des bibliothèques internes, elle comprend, en plus des statistiques, toutes les données qui ont été introduites lors de la création de la bibliothèque en tant qu'Info sur le Projet, ce qui englobe entre autres les 'Informations sur l'attribution d'une licence'. Dans le cas des bibliothèques externes, elle indique le nom de la bibliothèque et le chemin d'accès.

6.5 Journal

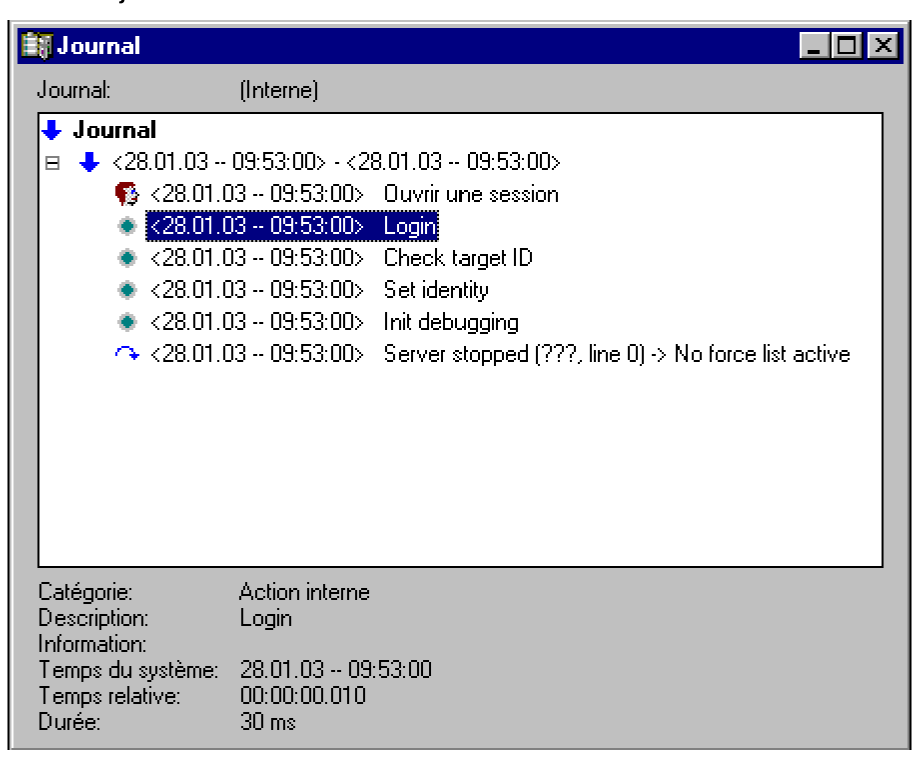
Le journal enregistre dans l'ordre chronologique toutes les actions qui ont lieu lors d'une session En Ligne. Pour ce faire, un fichier journal en format binaire (*.log) est créé pour chaque projet. En outre, l'utilisateur peut sauvegarder des extraits provenant de chaque journal de projet dans un journal externe.

Vous pouvez accéder à la fenêtre du journal En ligne et Hors ligne, et celle-ci peut ainsi également servir d'espion direct en mode En ligne.

'Fenêtre' 'Protocole en ligne'

Pour y avoir accès, sélectionnez l'option 'Fenêtre' 'Protocole en ligne' ou encore l'entrée correspondante ('Journal') dans l'onglet Ressources.

Fenêtre de journal



Le nom du journal en cours d'utilisation est indiqué à la suite de **Journal :** au dessus de la fenêtre de protocole. S'il s'agit du journal du projet en cours, "(Interne)" est également affiché.

Les entrées consignées sont indiquées dans la fenêtre de protocole. La dernière entrée est toujours ajoutée à la fin.

Les actions des catégories activées dans le cadre du menu 'Projet' 'Options' 'Journal', rubrique 'Filtre' sont seules visualisées.

Les informations relatives à l'entrée sélectionnée dans la fenêtre de protocole sont affichées en dessous de cette même fenêtre.

Catégorie : Il s'agit de la catégorie de l'entrée de journal. Les quatre catégories suivantes sont possibles :

- **Actions par l'utilisateur** : l'utilisateur a exécuté une action En ligne (normalement à partir du menu En ligne).
- **Actions internes** : une action interne a été exécutée au niveau En ligne (par exemple *Effacer mémoires tampons (Delete Buffers)* ou *Init débogage (Init Debugging)*).
- **Changements de l'état** : le statut du système d'exécution a été modifié (par exemple, passage de *En cours* (*Running*) à *Pause* (*Break*) si on est passé par un point d'arrêt).

- **Exceptions** : Une exception est survenue, par exemple une erreur de communication

Description : Il s'agit du type d'action. Les actions de l'utilisateur portent le même nom que le menu correspondant, toutes les autres actions sont libellées en anglais et portent le même nom que leur fonction `OnlineXXX()` correspondante.

Information : Ce champ contient une description des erreurs qui se sont éventuellement produites au cours de l'action. S'il n'y a pas eu d'erreur, ce champ reste vide.

Temps du système : Le temps du système au début de l'action, à la seconde près.

Temps relatif : le temps par rapport au début de la session En ligne, à la milliseconde près.

Durée : la durée de l'action en millisecondes.

Menu Journal

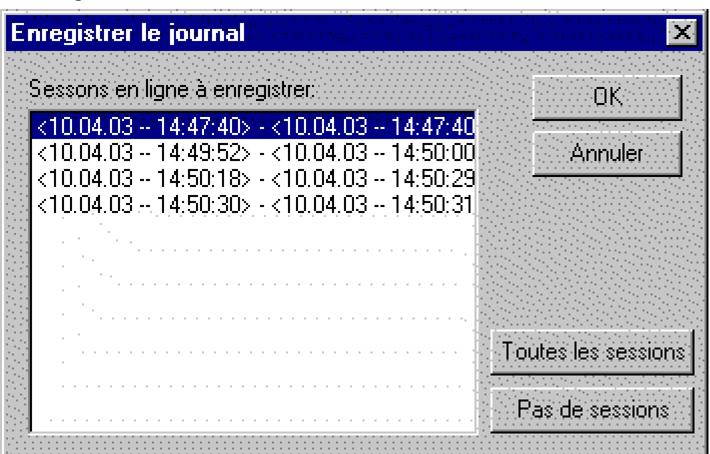
Si la fenêtre du journal est la fenêtre active, l'option **Journal** sera affichée à la place du point 'Insérer' dans la barre des menus.

Le menu vous offre le choix entre les points suivants :

Charger… Vous pouvez charger et visualiser un fichier de journal externe * log par le biais de la boîte de dialogue standard permettant d'ouvrir un fichier.

Le journal se trouvant dans le projet ne sera pas écrasé par cette commande. Si vous refermez la fenêtre du journal et que vous l'ouvrez à nouveau tout de suite ou débutez une nouvelle session En ligne, la version chargée sera à nouveau remplacée par le journal du projet.

Enregistrer… Cette option ne peut être sélectionnée que si à cet instant le journal du projet est affiché. Cette commande permet de sauvegarder un extrait du journal du projet dans un fichier externe. À cet effet, la boîte de dialogue suivante s'affiche à l'écran, vous permettant de sélectionner les sessions En ligne à sauvegarder.



Après avoir effectué votre sélection, la boîte de dialogue standard pour la sauvegarde d'un fichier s'ouvre ('Enregistrer le journal').

Afficher le journal du projet Cette commande ne peut être sélectionnée que si à cet instant un journal externe est affiché. Elle permet de revenir à l'affichage du journal du projet.

Enregistrer le journal

Indépendamment d'une sauvegarde possible du journal dans un fichier externe, le journal du projet est enregistré automatiquement dans un fichier binaire <Nom du projet>.log. Si un autre chemin

d'accès n'est pas donné explicitement dans la boîte de dialogue 'Projet' 'Options' 'Journal', la création se fera dans le même répertoire que celui dans lequel le projet est enregistré.

Le nombre maximal de sessions En ligne pouvant être enregistrées peut être déterminé dans la boîte de dialogue 'Projet' 'Options' 'Journal'. Si ce nombre est dépassé durant l'enregistrement en cours, les sessions les plus anciennes sont effacées de la mémoire tampon au profit des nouvelles.

6.6 Configuration de l'automate

6.6.1 Apercu

La configuration de l'automate est présente sous forme d'objet dans l'onglet **Ressources** de l'Organisateur d'objets. Elle donne accès à un éditeur de configuration avec lequel le matériel cible, sur lequel le projet actuellement ouvert doit tourner, peut être décrit. Pour élaborer le programme, il importe de connaître le nombre et l'emplacement des différentes entrées et sorties. **CoDeSys** se fonde sur cette description pour vérifier si les adresses CEI utilisées dans le programme existent bien au niveau du matériel.

Les fichiers de configuration (* cfg, voir ci-dessous 'Remarques sur la compatibilité') ou les fichiers de matériel (p.ex. *.gsd, .eds) sont à la base de tout travail dans l'éditeur de configuration ; ces fichiers sont à trouver dans un répertoire défini par le fichier cible (voir Configuration du système cible) ou défini dans les options du projet et sont lus lors du démarrage du projet. Le fichier de configuration décrit une configuration de base qui est automatiquement visualisée lors de l'ouverture de l'éditeur et indique les possibilités de paramétrage encore disponibles dans l'éditeur.

Attention : Lorsque le fichier de configuration est modifié, la configuration se basant sur ce fichier doit être refaite dans CoDeSys.

Remarques sur la compatibilité : Un **nouveau format de configuration d'automate** a été introduit dans CoDeSys V2.2, et les fichiers de configuration ayant servi de base sont munis de l'extension ***.cfg**. Le configurateur d'automate utilisé pour la programmation de projets plus anciens se basait quant à lui sur des fichiers de configuration reconnaissables à l'extension ***.con**. Vous pouvez déterminer dans le fichier cible si le 'vieux' configurateur doit continuer à être utilisé, même si un projet plus ancien est ouvert en V2.2 ou plus. Ceci vous épargne une transcription des fichiers de configuration vu que les fichiers *con peuvent être utilisés tout en restant inchangés. Au cas où cette définition manque dans le fichier cible, la configuration de l'automate qui est enregistrée dans le projet peut être convertie au nouveau format, pour autant qu'un fichier *.cfg ait été préparé à cet effet. Voir pour ce faire 'Extras' 'Convertir'.

Le configurateur d'automate CoDeSys permet la connexion de modules d'E/S simples ainsi que de modules Profibus et CAN. Après que l'utilisateur ait terminé la configuration dans l'éditeur, une représentation binaire du projet est transmise à l'automate lors du téléchargement de ce projet.

Configuration d'automate avec modules CAN et boîte de dialogue des réglages généraux

Configuration de l'automate
PLC Configuration
Word1 AT %IW0: WORD; (* First Word Input or Output
Module Type I/O[SLOT]
InByte1 AT %IB0.0: BYTE; (* First In Byte *) [CHAN
AT %IX0.0.0: BOOL; (* Bit 0 *)
AT %IX0.0.1: BOOL; (* Bit 1 *)
AT %IX0.0.2: BOOL; (* Bit 2 *)
AT %IX0.0.3: BOOL; (* Bit 3 *)
AT %IX0.0.4: BOOL; (* Bit 4 *)
AT %IX0.0.5: BOOL; (* Bit 5 *)
AT %IX0.0.6: BOOL; (* Bit 6 *)
AT %IX0.0.7: BOOL; (* Bit 7 *)
8 Byte digital Input[SLOT]
Module Typ CAN, CANMaster[SLOT]
Settings | Paramètre de module
Calculer les adresses automatiquement:
Vérifier superposition d'adresses:

La configuration est présentée dans l'éditeur sous forme d'arborescence et peut être traitée par le biais de commandes de menus et de boîtes de dialogue. Il existe des éléments qui sont soit des entrées, soit des sorties, soit des éléments de gestion, possédant eux-mêmes des sous-éléments (p.ex. un bus CAN, un PROFIBUS ou une carte d'entrées digitales à 8 entrées).

Les entrées et sorties apparaissent dans l'éditeur avec l'adresse CEI permettant d'y accéder. Dans les cas standard, on peut attribuer à chaque entrée ou sortie un nom symbolique permettant de l'identifier ; celui-ci sera placé avant l'adresse CEI.

Si vous ouvrez des projets contenant une configuration d'automate effectuée à l'aide des versions plus anciennes de CoDeSys, celle-ci peut être convertie au nouveau format de configuration standard d'automates.

La fenêtre de l'éditeur de configuration est divisée en deux : la partie gauche contient l'arborescence de configuration. La structure et le contenu de cette arborescence se composent en premier lieu des définitions des fichiers de configuration (configuration par défaut), ils peuvent cependant être modifiés par le biais des configurations effectuées par l'utilisateur dans le cadre du projet. La partie droite de l'écran est réservée aux boîtes de dialogue de configuration correspondant aux éléments sélectionnés, réparties sur un ou plusieurs onglets.

L'affichage de la boîte de dialogue de configuration est activé par défaut mais peut être désactivé via la commande 'Extras' 'Propriétés'.

Le module "Root" est placé à la tête de l'arborescence de configuration, caractérisé par le symbole et une identification qui lui aura été attribuée dans le fichier de configuration. Les autres éléments de la configuration sont indentés en dessous et par ordre hiérarchique : il s'agit de modules de différents types (CAN, PROFIBUS, E/S), de canaux ou de canaux binaires (identifications de bits à l'intérieur d'un canal).

6.6.2 Travailler dans la Configuration de l'automate

Sélection d'éléments

Pour sélectionner des éléments, cliquez avec la souris sur l'élément correspondant ou déplacez le rectangle en pointillés sur l'élément voulu, au moyen des touches directionnelles.

Les éléments qui sont précédés d'un signe plus sont des éléments d'organisation contenant des sous-éléments. Pour ouvrir un tel élément, il faut le sélectionner et double-cliquer sur le signe plus ou encore appuyer sur la touche <Entrée>. Les éléments ouverts (signe moins devant l'élément) se laissent refermer de la même manière.

Insérer les éléments, 'Insérer' 'Insérer élément', 'Insérer' 'Ajouter sub-élément'

Certains éléments de configuration d'automate sont déjà disponibles dans l'arborescence de configuration à l'ouverture du projet, et ils sont fonction des données par défaut du (des) fichier(s) de configuration d'automate et des fichiers d'outils. Si un de ces éléments disponibles est sélectionné, d'autres éléments peuvent être également insérés en fonction des définitions du fichier de configuration d'automate et de la présence des fichiers d'outils nécessaires. Différentes commandes vous sont proposées à cet effet :

- le menu 'Insérer' 'Insérer élément' : un élément peut être sélectionné et inséré avant l'élément marqué.
- le menu 'Insérer' 'Ajouter sub-élément' : un élément peut être sélectionné et inséré comme dernier sous-élément de l'élément marqué.

Vous trouvez toutes les commandes les plus importantes dans le menu contextuel (touche droite de la souris).

Remplacer des éléments, 'Remplacer élément'

Si la définition du fichier de configuration le permet, vous pouvez remplacer un élément sélectionné dans l'arborescence de configuration par un autre élément. Ceci comprend également la permutation de canaux configurés de telle sorte qu'ils soient utilisés comme entrée ou comme sortie. Utilisez la commande 'Extras' 'Remplacer élément'.

Attribution d'un nom symbolique

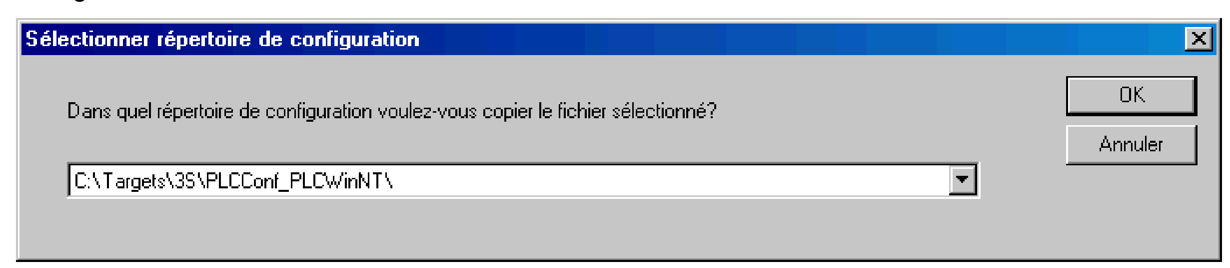
Vous avez déjà pu attribuer des noms symboliques pour les modules et canaux dans le fichier de configuration. Dans l'arborescence de configuration, ces noms apparaissent avant l''AT' de l'adresse CEI se rapportant à l'élément concerné. Le fichier de configuration détermine également si un nom symbolique peut être édité ou être attribué dans l'éditeur de configuration. Pour une nouvelle attribution, vous pouvez ouvrir un champ de saisie en cliquant sur l''AT' de l'adresse CEI, non sans avoir préalablement sélectionné un élément. De la même manière, vous pouvez éditer un nom symbolique déjà disponible en cliquant sur celui-ci. Veillez à ce que le nom symbolique attribué corresponde aux conventions des variables du projet !

Ajouter fichier de configuration

Cette commande du menu 'Extras' vous permet d'ajouter un fichier supplémentaire aux Fichiers de configuration. Sous fichiers de configuration, on comprend ici tous les fichiers contenus dans les répertoires de "Fichiers de configuration" définis dans les Options de projet.

Le dialogue **Sélectionner fichier de configuration** s'ouvre, permettant de définir un filtre (type de fichier) pour les fichiers CAN (*.eds,*. dcf), Profibus (gsd*.*), et les fichiers de configuration (*.cfg), ou encore pour tous les fichiers (*.*). Après la sélection du fichier souhaité, on vérifie si ce fichier se trouve déjà dans un des répertoires définis pour les fichiers de configuration. Un message approprié s'affiche si tel est le cas, et le fichier ne peut dès lors être ajouté. Si vous avez sélectionné un fichier .cfg, vous obtenez en outre une remarque reprenant ce dont il faut tenir compte dans un tel cas.

Si le fichier peut être ajouté, vous avez accès au dialogue **Sélectionner répertoire de configuration** proposant tous les répertoires définis pour le projet. Sélectionnez le répertoire dans lequel le fichier doit être copié. Après avoir confirmé votre sélection, ce fichier est directement disponible dans la configuration de l'automate.



Recalcul des adresses de modules, 'Extras' 'Calcul des adresses'

Pour autant que l'option 'Adresses automatiques' ait été activée dans le cadre des réglages généraux de la configuration d'automates, vous pouvez procéder à un nouveau calcul d'adresse par le biais de la commande 'Extras' 'Calcul des adresses', comprenant tous les éléments suivant le module sélectionné.

Retour à la configuration par défaut, 'Extras' 'Configuration par défaut'

Après des modifications dans l'éditeur de configuration, vous pouvez, au moyen de la commande 'Extras' 'Configuration par défaut', rétablir la configuration originale par défaut de l'automate, enregistrée dans le projet et se basant sur le fichier de configuration.

Attention: Dans le fichier de configuration, on peut grâce à une entrée déterminer que la configuration par défaut soit **toujours** restaurée lors de l'ouverture d'un projet. Ainsi, toutes les adaptations entreprises au sein de l'éditeur de configuration sont perdues !

Conversion d'anciens fichiers de configuration, 'Extras' 'Convertir'

Cette commande est disponible dans le menu 'Extras' lorsque vous ouvrez un projet pour lequel une configuration d'automate a été créée dans une version CoDeSys antérieure à V2.2 et pour autant qu'il ne soit pas précisé dans le fichier cible que le configurateur utilisé à l'époque doive encore être utilisé. Lorsque tous les fichiers de configuration nécessaires sont disponibles **(Attention : les informations dans les fichiers *.con doivent maintenant être contenues dans un fichier de configuration *.cfg !)**, vous pouvez les convertir à la configuration standard d'automates par le biais de la commande 'Convertir'. Vous obtenez pour ce faire une boîte de dialogue vous demandant : "Convertir la configuration d'automate au nouveau format? Attention : la conversion se fait à titre définitif !" ; vous devez alors fermer cette boîte de dialogue en répondant par Oui on Non. Si vous optez pour 'Oui', l'éditeur de configuration d'automate se referme et affiche le nouveau format lors de toute nouvelle ouverture.

Attention: L'ancienne configuration ne peut plus être rétablie après une conversion.

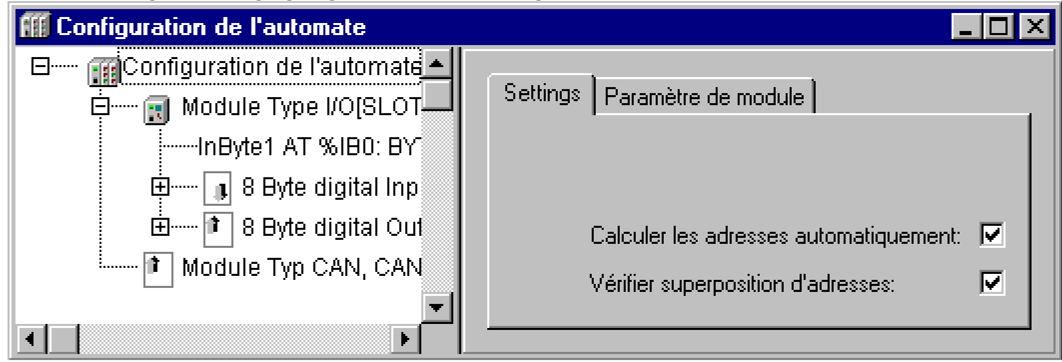
6.6.3 Settings

Marquez l'entrée en tête de l'arborescence de configuration (module Root). Vous obtenez alors la boîte de dialogue 'Settings' :

Calculer les adresses automatiquement: Chaque module que vous ajoutez obtient automatiquement une adresse qui est le résultat du module inséré juste avant, ajouté à sa propre dimension. Si vous enlevez un module de la configuration, les adresses des modules suivant celui-ci sont automatiquement adaptées en conséquence. La commande 'Extras' 'Calcul des adresses' permet de recalculer toutes les adresses à partir du nœud (module) actuellement sélectionné.

Vérifier superposition d'adresses: Les superpositions d'adresses sont vérifiées et communiquées lors de la compilation du projet.

Boîte de dialogue des réglages généraux de la configuration d'automates



Le mode global d’attribution d’adresses (adresses plates / adresses selon l’identité) au sein de la configuration d’automate est prédéfini dans le fichier de configuration.

6.6.4 Boîte de dialogue de paramétrage spécifique à l'application (Custom Parameters)

Grâce à une DLL spécifique à l’application (dialogue individuel), vous pouvez élargir les possibilités de paramétrage au sein du configurateur. Cette DLL 'Hook' est créée dans le même répertoire que le fichier de configuration et y est rattachée par le biais d’une entrée lors de la description des classes de modules ou de canaux. Pour le module concerné ou les canaux s’y rapportant, on obtient la boîte de dialogue définie dans la DLL en lieu et place de la boîte de dialogue standard 'Paramètres de module'

Exemple d'une boîte de dialogue de paramétrage spécifique à l'application (Custom Parameters)

Basisparameter | Custom Parameters

Custom Dialog

PgmVersion: 0
LibDir: d:\codesys\lib
ObjectId: 1
SectionName: Module.CPU1
PrmCount: 9

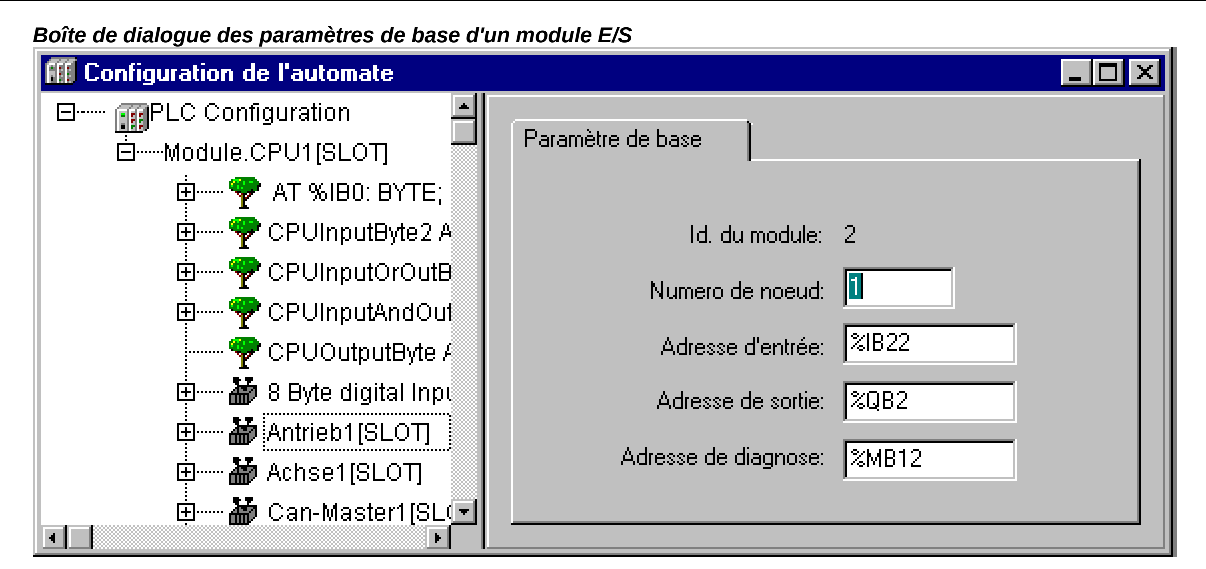
Definierte Parameter
1, 10000, Parameter.RateType, XRate, 10
2, 10000, Parameter.RateType, Yrate, 10
3, 10001, Parameter.BoolType, EnableDiags, Yes
4, 10007, Parameter.NameType, Namen, HugoType
5, 107659, Parameter.FileType, DumpFile, D:\CoDeSys\Projekte\Dump.bin
6, 1343, Parameter.NameString, IDString, abc def
7, 1347, Parameter.RealType, Realvalue, 1.876
8, 1348, Parameter.Bool, Boolvalue, TRUE
9, 138740, Parameter.IntTypeHex, Test, 16#A

Param1 (Value): 10

Paramètrage d'un module E/S

6.6.5 Paramètres de base d'un module E/S

Boîte de dialogue des paramètres de base d'un module E/S



Si vous sélectionnez un module d'E/S dans l'arborescence de configuration, vous obtenez la boîte de dialogue des 'Paramètres de base' comprenant les entrées suivantes :

Id. du module : l'identité du module est une identification univoque du module au sein de l'environnement total de configuration. Elle est reprise hors du fichier de configuration et n'est pas éditable.

Numéro de noeud : le numéro de noeud découle d'une entrée dans le fichier de configuration ou de sa position dans l'arborescence de configuration si une telle entrée n'est pas définie dans le fichier de configuration.

Adresse d'entrée, Adresse de sortie, Adresse de diagnose : adresses pour les entrées et sorties ou pour l'enregistrement des données de diagnostic.

Ces adresses se rapportent au module. Les adresses déjà définies, les modes d'adressages utilisés dans la configuration et si cet emplacement est encore éditable dépendent des réglages généraux (Settings) et des définitions dans le fichier de configuration.

Ne pas changer les adresses automatiquement :cette option n'est disponible que si cela a été défini dans le fichier de configuration. Si elle est activée, le module est exclu lors d'un Calcul automatique des adresses.

Charger description du module : Si cette option est désactivée (elle l'est par défaut), le module ne sera pas pris en compte lors d'un téléchargement. On définit dans le fichier de configuration *.cfg si l'option doit être visible et éditable dans le présent dialogue.

Vous trouverez de plus amples informations relatives au diagnostic dans la configuration d'automate :

Une adresse de mémento doit être saisie dans le champ **Adresse de diagnostic**, à laquelle les informations de diagnostic sont automatiquement écrites. Dans le cas des modules d'E/S normaux, la façon dont l'adresse de diagnostic est traitée dépend de la configuration spécifique du matériel. Pour les systèmes de Bus CAN ou Profibus DP, les adresses de diagnostic sont soumises aux points suivants : Il faut spécifier une adresse mémento à partir de laquelle se fera l'enregistrement des informations de diagnostic. Ces informations sont créées dans une structure *GetBusState* contenue dans une bibliothèque spécifique au fabricant (par exemple BusDiag.lib de 3S - Smart Software Solutions GmbH).

Après qu'une tâche CEI ait envoyé ses données de processus aux module E/S ou lu ces données à partir des modules, tous les modules sont régulièrement invité à remplir la structure de diagnostic *GetBusState*.

Si un des participants disponibles sur le bus annonce une erreur, ses données spécifiques de diagnostic peuvent être lues avec le module *DiagGetState* (également défini dans la bibliothèque ci avant nommée). Cette fonction n'est cependant active que si le Bus principal a également été configuré dans le cadre de CoDeSys.

Voyez ci-dessous les paramètres d'entrée et de sortie pour le bloc fonctionnel *DiagGetState*, appelé pour la lecture des informations de diagnostic pour un participant de Bus précis.

Variables d'entrée du *DiagGetState*:

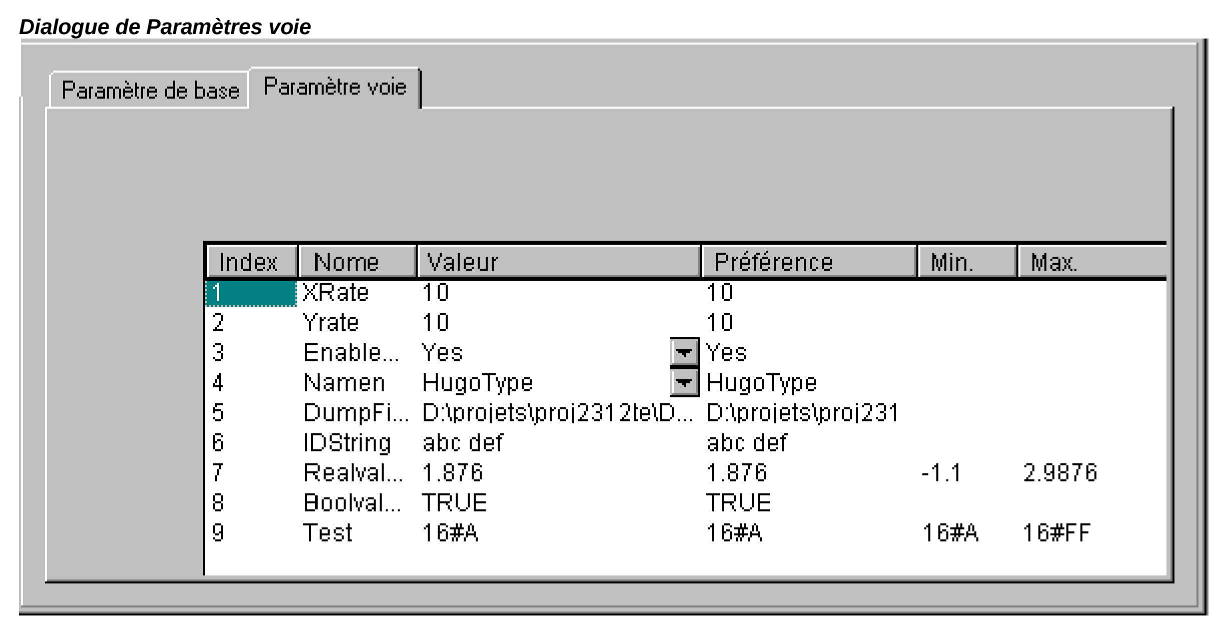
ENABLE: BOOL;	L'exécution du module débute avec un front montant
DRIVERNAME: POINTER TO STRING;	Nom du pilote (adresse du nom) vers lequel l'ordre de diagnostic doit être envoyé. Si vous introduisez 0, l'ordre de diagnostic sera envoyé à tous les pilotes disponibles.
DEVICENUMBER:INT;	Identification du bus qui est géré à partir de ce module (pilote). À titre d'exemple, le pilote Hilscher peut gérer jusqu'à 5 cartes (bus). L'index commence à zéro.
BUSMEMBERID: DWORD;	Identification univoque et spécifique au bus / au module du participant de bus. (p.ex. NodeID dans le cas d'un carte CANopen, adresse du participant dans le cas d'une carte PB-DP)

Variables de sortie du DiagGetState :

READY: BOOL ;	TRUE : le traitement de l'ordre de diagnostic est terminé.
STATE:INT;	Si READY = TRUE, STATE donne des renseignements sur le statut actuel du module au moyen d'une des valeurs ci-dessous. Elles sont affectées à des constantes globales : -1: paramètre d'entrée incorrect (NDSTATE_INVALID_INPUTPARAM:INT;) 0: le module ne fonctionne pas (NDSTATE_NOTENABLED:INT;) 1: le module lit les informations de diagnostic (NDSTATE_GETDIAG_INFO:INT;) 2: les informations de diagnostic sont maintenant disponibles (NDSTATE_DIAGINFO_AVAILABLE:INT;) 3: pas d'informations de diagnostic disponibles (NDSTATE_DIAGINFO_NOTAVAILABLE:INT;)
EXTENDEDINFO: ARRAY [0..129] OF BYTE;	Jusqu'à 100 octets pour les données de diagnostic du participant de bus, spécifiques au fabricant. Un octet est réservé pour chaque participant de bus possible, et les bits 0 - 2 sont utilisés comme suit : Bit 0 : le participant de bus est disponible dans la configuration. Bit 1 : le participant de bus est disponible sur le bus. Bit 2 : le participant de bus indique une erreur.

6.6.6 Paramètres voie / Customer paramètres d'un module E/S

Dialogue de Paramètres voie



Les paramètres donnés dans le fichier d'appareils sont visualisés. On ne peut procéder à l'édition que dans la colonne Valeur.

Index : il s'agit d'un numéro courant (i) qui numérote les paramètres au sein du module.

Nom : nom du paramètre

Valeur : valeur modifiable du paramètre

La valeur par défaut est tout d'abord affichée. Les valeurs peuvent être affichées directement ou par le biais d'un nom symbolique. Elles peuvent être éditées, pour autant que les entrées du fichier de configuration ne soient pas réglées sur 'Lecture seule', en ouvrant un champ de saisie au moyen d'un clic sur la valeur ou en sélectionnant une autre valeur dans une liste de défilement. Si la valeur mentionne un fichier, un double-clique sur cette valeur ouvre une boîte de dialogue 'Ouvrir fichier' qui vous permet alors de sélectionner un autre fichier.

Préférence : valeur par défaut du paramètre

Min. : valeur minimale du paramètre (uniquement si représentation directe des valeurs)

Max. : valeur maximale du paramètre (uniquement si représentation directe des valeurs)

Le cas échéant, vous pouvez obtenir des informations supplémentaires sur le paramètre actuellement sélectionné par le biais d'une info-bulle.

La boîte de dialogue des paramètres spécifiques à l'application peut apparaître à la place de la boîte de dialogue des paramètres du module. C'est le cas lorsqu'une boîte de dialogue pour les paramètres spécifiques à l'application a été 'rattachée' pour le module concerné au moyen d'une DLL Hook.Paramètrage d'un canal

6.6.7 Paramètre de base d'un canal

Boîte de dialogue des paramètres de base d'un canal

Paramètre de base | Custom Parameters

Adresse: %IB0

Commentaire: First Input Byte

Id. de la voie: 1000

Classe: I

Taille: 8

Default identifier: CPUInputByte1

Commentaire : informations supplémentaires au sujet du canal.

Un commentaire prédéfini éventuel peut être édité ou un nouveau commentaire peut être saisi dans ce champ.

Id. de la voie : identification globale univoque du canal.

Classe : indique si le canal est utilisé comme entrée (I), comme sortie (Q) ou comme entrée et sortie (I&Q), ou encore s'il est permutable en tant que tel (I|Q). Si le canal est permutable, vous pouvez le faire par le biais de la commande 'Extras' 'Remplacer l'élément'.

Taille : taille du canal [octets]

Default identifier : nom symbolique du canal attribué dans le fichier de configuration.

Le nom du canal est attribué dans le fichier de configuration. Seulement si le module père est configuré en conséquence, il peut être édité dans l'arborescence de configuration.

Adresse : ce champ de saisie n'apparaît que lorsqu'il a été activé via une entrée dans le fichier de configuration. Vous pouvez introduire ici l'adresse du canal souhaité.

Paramètre voie d'un canal

Conformément à la boîte de dialogue des paramètres d'un module E/S, cette boîte de dialogue sert à la visualisation et à la modification des valeurs paramètres du canal : **Index, Nom, Valeur, Préférence, Min., Max..** Comme c'est le cas pour les modules, elle peut être remplacée par une boîte de dialogue 'Custom Parameters' spécifique à l'application.

Canaux binaires

Les canaux binaires sont automatiquement insérés lorsqu'un canal dispose dans le fichier de configuration de l'entrée CreateBitChannels=TRUE.

Les paramètres de base dans le cas des canaux binaires ne comprennent que le champ de saisie Commentaire.

6.6.8 Configuration de modules Profibus

CoDeSys repose sur une configuration de matériel conformément au standard PROFIBUS-DP. Il faut pour ce faire disposer d'un fichier de configuration qui permette l'insertion de modules Profibus.

Un système PROFIBUS-DP se compose d'un ou de plusieurs maître(s) et de ses (leurs) esclaves. Pour que les appareils puissent échanger des données entre eux via le bus, ils doivent tout d'abord être configurés. Lors de la mise en service suivant ces configurations, chaque maître paramètre les esclaves qui lui ont été attribués lors de la configuration. En fonctionnement, un maître envoie des données à ses esclaves respectifs et/ou exige de leur part des données.

La configuration des appareils maîtres et esclaves se base dans **CoDeSys** sur des fichiers GSD. Ces fichiers GSD (**G**eräte**S**tamm**D**aten = données principales des appareils) sont fournis par le fabricant des appareils et comprennent une description uniformisée des caractéristiques d'un appareil PROFIBUS-DP. Veillez à ce que les fichiers GSD nécessaires soient présents dans le répertoire défini pour les fichiers de configuration avant de démarrer le programme CoDeSys.

Les appareils adéquats peuvent alors être insérés dans l'arborescence de configuration via des dialogues et on peut procéder à l'adaptation de leurs paramètres. Un ou plusieurs esclave(s) peu(ven)t être rattaché(s) à un maître.

Si un DP maître est marqué dans l'arborescence de configuration, les dialogues suivants (choix en fonction de la définition au sein du fichier de configuration) peuvent être choisis par le biais de leur onglet : paramètres de base, paramètres DP, paramètres de bus, paramètres de module. Le second onglet porte éventuellement un autre titre que "Paramètres DP" défini dans le fichier de configuration.

Si un DP esclave rattaché à un DP maître est marqué, vous obtenez les dialogues suivants : paramètres de base, paramètres DP, entrées/sorties, paramètres d'utilisateur, assignation à un groupe, paramètres de module.

Par contre, si un DP esclave est configuré comme esclave au niveau maître, les dialogues suivants sont nécessaires à la configuration : paramètres de base, paramètres DP, entrées/sorties, paramètres de module.

Paramètre de base d'un DP-Maître

La boîte de dialogue des paramètres de base d’un DP maître correspond à celle des autres modules (voir chapitre 6.6.5, Paramètres de base d'un module E/S).

Paramètre DP d'un DP-Maître

Cette boîte de dialogue indique les paramètres ci-dessous du DP esclave, repris hors du fichier d'appareils (le dialogue porte éventuellement un autre titre défini dans le fichier de configuration).

Info — **Fabricant**, **révision** GSD, **numéro d'identification**, **version du matériel** et **version du logiciel**, **nom du fichier** (GSD)

Adresses — **Adresse de station** : la gamme possible va de 0 à 126. Chaque nouvel appareil inséré dans le bus se voit automatiquement attribuer l'adresse directement au dessus. (Veuillez noter : l'adresse 126 est l'adresse d'esclave par défaut). Une saisie manuelle est possible, et un contrôle est effectué à la recherche d'adresses attribuées deux fois.

Numéro de station la plus haute : L'adresse la plus haute (HSA) attribuée dans le bus est affichée. Vous pouvez saisir une adresse inférieure pour réduire la zone GAP (il s'agit de la gamme d'adresses qui est parcourue à la recherche de nouveaux appareils actifs).

Boîte de dialogue des paramètres DP pour DP maître

Paramètre de base | Paramètre DP | Paramètre de bus

Info
Fabricant: Hilscher GmbH
Révision: Version 1.000
Numéro d'identification: 0x0447
Version du matériel: Version 1.000
Version du logiciel: Version 1.000
Nom du fichier: Hil_0447.gsd

Fichier GSD...

Adresses
Adresse de station: 1
Numéro de station la plus haute: 1

Mode
Mode effacer automatiquement
Démarrer automatiquement

Groupes...

Le bouton **Fichier GSD** vous permet d'ouvrir et de visualiser le fichier GSD relatif aux appareils.

Le bouton **Groupes** vous donne accès à la boîte de dialogue 'Caractéristiques des groupes'. Les caractéristiques de groupe se rapportent aux esclaves affectés au maître. Vous pouvez aménager jusqu'à huit groupes. Déterminez pour chaque groupe s'il doit fonctionner en **mode Freeze** et/ou en **mode Sync**. Grâce au classement des esclaves en différents groupes (voir ci-dessous : 'Caractéristiques des DP esclaves', 'Assignation à un groupe'), l'échange de données peut être synchronisé via une commande de contrôle global à partir du maître. Par la commande *Freeze*, le maître ordonne à un esclave ou à un groupe de « geler » ses entrées dans leur état momentané et de transmettre ces données lors de l'échange de données suivant. Une commande Sync oblige les esclaves à transmettre aux sorties les données reçues par le maître lors de l'échange de données suivant et ce de manière synchrone lors de la prochaine commande Sync.

Pour (dés)activer les options Freeze et Sync d'un groupe, cliquez à l'aide du bouton gauche de la souris sur l'endroit correspondant du tableau de façon à placer/enlever un 'X' auprès de l'option souhaitée, tandis que le bouton droit de la souris vous permet d'activer ou de désactiver cette option via un menu contextuel. En outre, vous pouvez éditer les noms des groupes.

Paramètre DP du DP maître / Caractéristiques dau groupe

Caractéristiques du groupe

Nom du groupe	Sync. Mode	Freeze Mode
Gr 1		X
Gr 2	X	X
Gr 3		
Gr 4	X	
Gr 5	X	
Gr 6		X
Gr 7	X	X
Gr 8	X	X

OK | Annuler

Paramètre de bus d'un DP-Maître

Le jeu des paramètres de bus décrit le comportement de la communication dans le temps. Les valeurs des paramètres individuels sont automatiquement calculées sur base des données contenues dans les fichiers GSD et en fonction du **Débit en bauds** réglé par l'utilisateur, pour autant que l'option **Optimisation automatique** soit activée.

Attention : Les valeurs calculées automatiquement ne sont que des valeurs approximatives !

Paramètres de bus du DP maître



Tous les paramètres peuvent également être édités manuellement selon votre gré.

Taux en bauds [kBits/s]	Vous pouvez choisir entre les réglages effectués dans le cadre du fichier GSD ; vous ne pouvez cependant régler qu'un débit de transmission qui sera supporté par tous les esclaves
Optimiser automatiquement	Si cette option est activée, les réglages effectués dans le dialogue 'Paramètres de bus' sont optimisés sur base des données contenues dans les fichiers GSD; vous ne pouvez procéder à une édition de ces valeurs que si l'option est désactivée. **Attention :** les valeurs calculées automatiquement ne sont que des valeurs approximatives !
Slot Time	Il s'agit du temps maximum durant lequel le maître, après un télégramme d'appel, attend le premier signe d'un télégramme de réponse de la part de l'esclave.
Min.Station Delay	min. TSDR (en Tbit) : temps minimal de réaction après lequel un participant de bus peut répondre (min. 11 Tbit)
Max.Station Delay	max TSDR (en Tbit) : durée maximale accordée à l'esclave pour une réponse.
Quiet Time	TQUI (en Tbit): Temps d'inactivité qui doit être respecté lors d'une conversion de signaux NRZ (Non Return to Zero) en d'autres codages (temps de commutation pour le répéteur)
Target Rotation Time	TTR (en Tbit): durée de rotation nominale d'un jeton ; durée projetée durant laquelle le maître devrait recevoir un jeton. Elle est le résultat de la somme des temps de retenue du jeton par tous les maîtres du bus.
Gap Update Factor	Facteur G d'actualisation GAP : nombre de cycles de bus après lequel des nouvelles stations actives sont recherchées dans le GAP du maître (gamme d'adresses à partir de la propre adresse jusqu'à l'adresse du participant actif suivant).
Max. Retry Limit	Nombre maximal de nouvelles tentatives d'appel par le maître, lorsque ce dernier n'a pas reçu de réponse valable de la part de l'esclave.
Min. Esclave Interval	Temps entre deux cycles de bus, durant lequel un esclave peut traiter une requête du maître (base = 100 µs). La valeur introduite ici doit être fonction des données par défaut contenues dans les fichiers GSD des esclaves.
Poll Timeout	Temps maximal après lequel le demandeur (DP maître de niveau 2)

	doit aller chercher la réponse du maître lors d'une communication maître-maître (base = 1ms).
Data Control Time	Temps durant lequel le maître communique son état de fonctionnement à ses esclaves. En même temps, le maître vérifie si un échange de données effectives a lieu avec les esclaves et actualise la liste Data_Transfer_List.
Watchdog Time	Valeur de temps pour la surveillance de la réponse (keep alive). Le réglage n'est pas actuellement supporté (réglé sur 400ms).

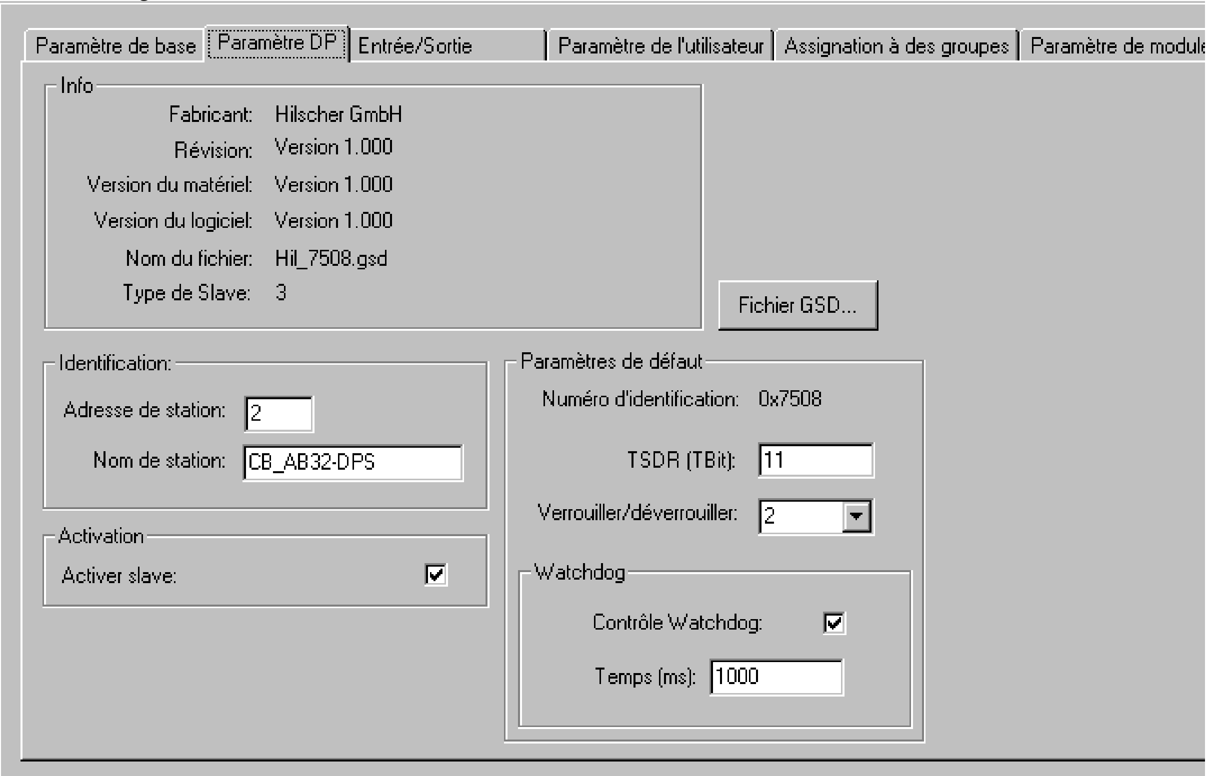
Paramètre de base d'un DP-Esclave

La boîte de dialogue des paramètres de base d'un DP esclave correspond à celle des autres modules (voir chapitre 6.6.5, Paramètres de base d'un module E/S).

Paramètre DP d'un DP-Esclave

Cette boîte de dialogue indique les paramètres ci-dessous du DP esclave, repris hors du fichier d'appareils (le dialogue porte éventuellement un autre titre défini dans le fichier de configuration):

Boîte de dialogue Paramètre DP d'un DP esclave



Info	**Fabricant**, **révision** GSD, **numéro d'identification**, **version du matériel** et **version du logiciel**, **nom du fichier** (GSD), **Type de slave**
Paramètres de défaut	**Numéro d'identification**: numéro univoque d'identification attribué par le PNO pour ce type d'appareil. Ceci permet une référence univoque entre le DP esclave et le fichier GSD correspondant **TSDR** (Tbit*): Time Station Delay Responder: temps de réaction après lequel l'esclave doit au plus tôt répondre au maître (min. 11 Tbit) * TBit: unité de temps pour la transmission d'un bit par PROFIBUS, inverse du débit de transmission ; par exemple 1 Tbit à 12MBaud=1/12.000.000 Bit/sec=83ns **Lock/Unlock**: l'esclave est bloqué ou libéré pour d'autres maîtres :

0: le TDSR min. et les paramètres spécifiques à l'esclave peuvent être écrasés ;
1: esclave libéré pour d'autres maîtres,
2: esclave bloqué pour d'autres maîtres, tous les paramètres sont repris ;
3: esclave à nouveau libéré pour d'autres maîtres

Identification Adresse de station (voir ‚Paramètres DP du maître DP‘), Nom de station (correspond au nom de l'appareil, éditable)

Activation l'esclave est actif/non actif dans la configuration actuelle. Si vous n'optez pas pour l'activation, les données de configuration des esclaves seront transmises au coupleur lors d'un téléchargement, mais un échange de données via le bus ne peut se produire.

Watchdog (surveillance) Si Contrôle Watchdog est activé, la durée introduite à cet effet vaut (surveillance de la réponse, base = 10 ms). Si le maître ne s'adresse pas à l'esclave durant ce temps, ce dernier revient à son état d'initialisation.

Entrée/Sorties d'un DP-Esclave

Boîte de dialogue pour la configuration des entrées / sorties d'un DP esclave

Paramètre de base | Paramètre DP | Entrée/Sortie | Paramètre de l'utilisateur | Assignation à des groupes | Paramètre de

Longueur max. de l'entrée::	128	Byte	Longueur de l'entrée:	4	Byte
Longueur max. de la sortie:	128	Byte	Longueur de la sortie:	0	Byte
Longueur max. de l'entrée/de la sortie:	256	Byte	Longueur de l'entrée/sortie:	4	Byte
No. max. de modules:	63		Numéro de modules:	1	

Modules d'entrée
- 6ES7 131-4BB00-0AB0 2DI DC24V
- 6ES7 131-4EB00-0AB0 2DI 120V
- 6ES7 131-4FB00-0AB0 2DI 230V
- 6ES7 134-4FB00-0AB0 2AI U
- 6ES7 134-4GB00-0AB0 2AI I 2DMU
- 6ES7 134-4GB10-0AB0 2AI I 4DMU
- 6ES7 134-4JB00-0AB0 AI 2xTC
- 6ES7 134-4JB50-0AB0 2AI RTD

Modules de sortie
Modules d'entrée et sortie
Modules vides

>> | Effacer | Caractéristiques

Modules sélectionnés
- 6ES7 134-4GB10-0AB0 2AI I 4DMU

La procédure à suivre pour la configuration de l'esclave est différente selon qu'il s'agisse d'un esclave 'modulaire' ou d'un esclave non modulaire 'fixe'.

La sélection des modules pour **l'esclave modulaire** s'effectue comme suit :

Vous sélectionnez le module d'entrée ou de sortie que vous souhaitez en cliquant dessus dans la liste de gauche et vous le copiez dans la fenêtre de droite par le biais du bouton **>>**. Des entrées incorrectes peuvent être corrigées en sélectionnant le module non souhaité dans la fenêtre de droite et en appuyant sur le bouton **Effacer**. Les modules insérés sont immédiatement repris dans l'arborescence de configuration. S'ils y sont marqués, vous avez accès au dialogue adéquat **Module Profibus**, qui reprend les adresses d'entrée, de sortie et de diagnostic du module. Si un canal appartenant à ce module est marquée, vous avez accès au dialogue **Canal Profibus** qui affiche

l'adresse du canal. Vous pouvez définir pour ces deux dialogues un autre titre dans le fichier de configuration.

Comme l'on doit tenir compte de la longueur maximale des données (**Longueur max. de l'entrée, Longueur max. de la sortie, Longueur max. de l'entrée / de la sortie**) et du nombre maximal de modules (**No. max. de modules)** indiqués dans le fichier GSD, ces informations sont affichées dans les deux listes de modules. Le bloc de gauche représente les valeurs maximales possibles pour l'appareil, le bloc de droite représente la somme des valeurs atteintes par la configuration sélectionnée. En cas de dépassement des valeurs maximales, un message d'erreur s'affiche.

La boîte de dialogue énumère dans la fenêtre de gauche tous les modules d'entrée et de sortie disponibles dans le fichier GSD de l'esclave tandis que la fenêtre de droite contient la configuration sélectionnée actuellement pour cet appareil en matière d'entrées et de sorties.

S'il s'agit d'un esclave modulaire (appareil qui peut être équipé de différents modules d'entrée et de sortie), la sélection s'effectue comme suit : vous sélectionnez le module d'entrée ou de sortie que vous souhaitez en cliquant dessus dans la liste de gauche et vous le copiez dans la fenêtre de droite par le biais de la touche **>>**. Des entrées incorrectes peuvent être corrigées en sélectionnant le module non souhaité dans la fenêtre de droite et en appuyant sur le bouton **Effacer**.

Ce mode de sélection n'est pas possible auprès des **esclaves non modulaires**. Ceux-ci imposent immédiatement une représentation fixe de leurs entrées et sorties dans la fenêtre de droite.

Le bouton **Caractéristiques** vous donne accès à la boîte de dialogue 'Caractéristiques du module' relative au module d'entrée ou de sortie actuellement sélectionné dans la liste de gauche ou de droite. Celle-ci indique le **Nom**, la **Configuration** (codification de la description du module selon la norme PROFIBUS) et la **Longueur de l'entrée (octets)** ainsi que la **Longueur de la sortie (Octets)** du module en **Octets**. Si, dans le fichier GSD, la description du module comprend, en plus du jeu standard de paramètres, des paramètres spécifiques, ceux-ci sont énumérés avec leur valeur et leur plage de valeur. Si vous activez l'option **Noms symboliques**, les noms symboliques seront utilisés.

Boîte de dialogue pour la caractéristiques de entrées/sorties d'un DP-Esclave

Caractéristiques du module

Nom 6ES7 134-4GB10-0AB0 2AI I 4DMU

Configuration: 0x51

Longueur de l'entrée (octets): 4

Longueur de la sortie (Octets): 0

Noms symboliques: ☑

OK

Annuler

Paramètre	Valeur	
"Sammeldiagnose"	sperren	
"Diagnose: Ueberlauf/Unterlauf"	sperren	
"Diagnose: Drahtbruch E0"	sperren	
"Diagnose: Drahtbruch E1"	sperren	
"Glaettung E0"	keine	
"Glaettung E1"	keine	
"Messart/-bereich E0"	Strom (4-DMU)	4 .. 20 mA
"Messart/-bereich E1"	Strom (4-DMU)	4 .. 20 mA

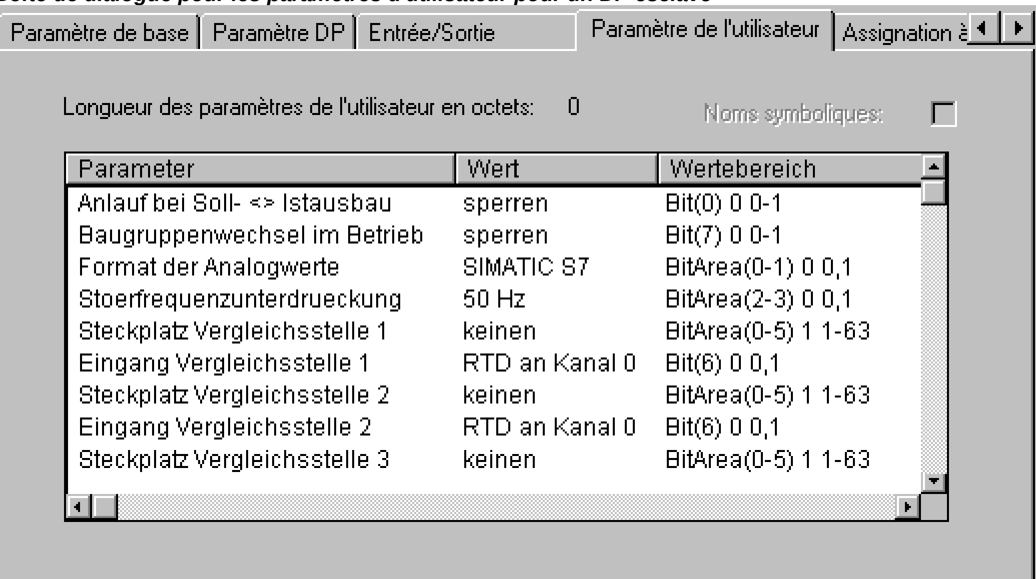
Paramètre de l'utilisateur d'un DP-Esclave

Différents paramètres étendus et définis dans le fichier GSD pour un DP esclave sont énumérés ici. La colonne Paramètre vous donne le nom du paramètre. Les valeurs de paramètres introduites dans la colonne Valeur peuvent être modifiées par un double-clic ou via le bouton droit de la souris. En outre, la plage de valeur est affichée.

Si des noms symboliques ont été attribués aux paramètres dans le fichier GSD, l'option **Noms symboliques** peut être activée de façon à ce que les valeurs soient affichées avec ces derniers. À

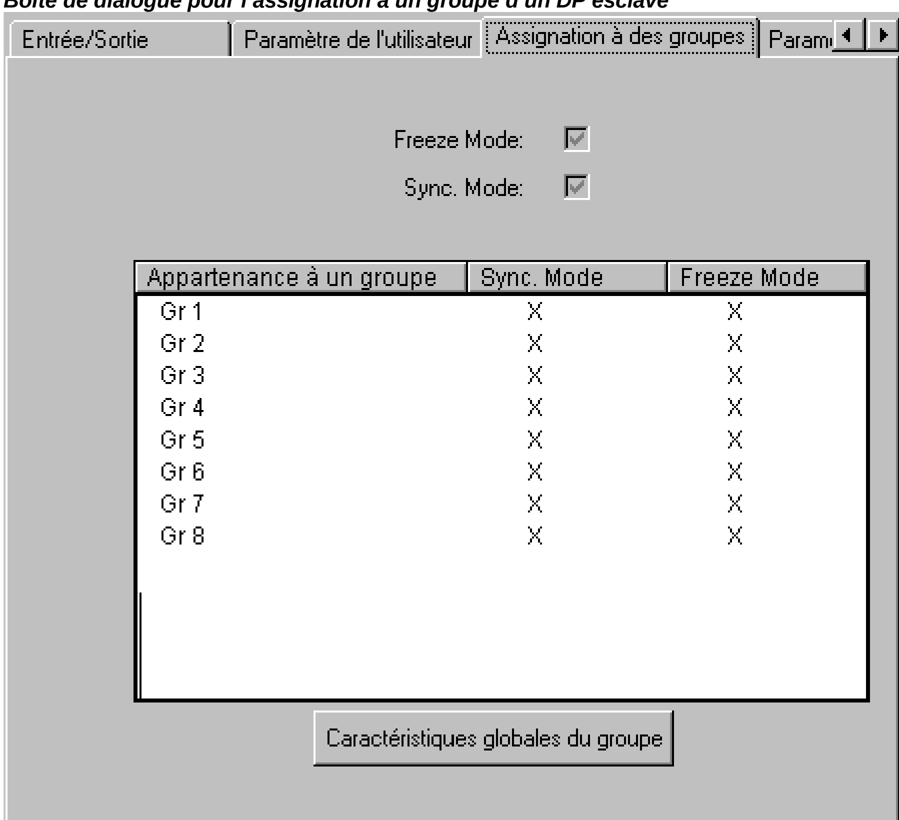
titre d'information, le tableau indique en outre la **longueur des paramètres de l'utilisateur en octets**.

Boîte de dialogue pour les paramètres d'utilisateur pour un DP esclave



Assignation à des groupes d'un DP-Esclave

Boîte de dialogue pour l'assignation à un groupe d'un DP esclave



Cette boîte de dialogue sert à assigner l'esclave à un ou plusieurs des huit groupes possibles. Les caractéristiques des groupes (**Mode Sync** et/ou **Mode Freeze**) généralement acceptées à cet égard sont définies lors de la configuration des caractéristiques du maître (voir 'Caractéristiques du DP maître', 'Caractéristiques des groupes'). On accède également à cette boîte de dialogue par le biais du bouton **Caractéristiques globales du groupe**.

Le(s) groupe(s) auxquel(s) l'esclave est assigné est (sont) marqué(s) d'un signe plus. L'assignation ou l'écartement d'un esclave par rapport au groupe s'effectue en sélectionnant le nom du groupe dans la colonne **Appartenance à un groupe**, puis en sélectionnant 'Ajouter un esclave au groupe' ou 'Écarter un esclave du groupe' par le biais du bouton droit de la souris, ou encore en cliquant à nouveau à gauche du nom du groupe.

Un appareil esclave ne peut être assigné qu'aux groupes dont il supporte les caractéristiques. Les caractéristiques de chaque esclave à ce sujet (**Mode Sync / Mode Freeze**) sont indiquées dans la partie supérieure du tableau. Les modes supportés par l'appareil sont munis d'un crochet.

Paramètre de module d'un DP-Esclave

La boîte de dialogue des paramètres de module d'un DP esclave correspond à celle des autres modules: Les paramètres qui ont été attribués à l'esclave en plus des paramètres DP et d'utilisateur dans le fichier de configuration sont représentés ici, et leurs valeurs peuvent normalement être éditées.

Caractéristiques d'un DP esclave lors d'un fonctionnement esclave du Profibus

Si le Profibus est exploité en mode esclave, l'appareil esclave est rattaché au niveau du maître. La configuration est possible par le biais de quatre onglets :

- Paramètres de base
- Paramètres DP
- Paramètres de module
- Entrées / Sorties

6.6.9 Configuration CAN

CoDeSys supporte une configuration de matériel conforme au projet de norme CANopen 301. Il faut pour ce faire disposer d'un fichier de configuration qui permette l'insertion de modules CAN.

Les fichiers EDS (**E**lectronic **D**ata **S**heet) ou DCF (**D**evice **C**onfiguration **F**ile) créés dans le répertoire destiné aux fichiers de configuration peuvent être utilisés lors de la configuration dans les limites de leur définition dans le fichier de configuration. Les possibilités de configuration d'un module CAN sont définies dans un fichier EDS. Si un module défini dans un fichier DCF est inséré, alors seules les adresses CEI peuvent être modifiées, étant donné que ce module à déjà été entièrement configuré dans un configurateur CAN.

Les modules CAN obtiennent une configuration qui permet de décrire leur comportement eu égard au temps et aux erreurs lors de l'échange d'informations (boîte de dialogue des paramètres CAN d'un module CAN). En plus, un mappage des PDO (Process Data Objects) est défini pour chaque module, servant à l'envoi ou à la réception (boîte de dialogue Recevoir et Envoyer mappage PDO). Les valeurs des SDO (Service Data Objects) disponibles peuvent être adaptées (boîte de dialogue Service Data Objects).

Grâce à la boîte de dialogue des paramètres de module, vous pouvez configurer les paramètres supplémentaires d'un module CAN ou d'un maître CAN, indiqués dans le fichier d'outils.

Paramètres de base d'un CAN-Maître

La boîte de dialogue des paramètres de base d'un maître CAN correspond à celle des autres modules (voir chapitre 6.6.5, Paramètres de base d'un module E/S).

Paramètre CAN d'un CAN-Maître

Boîte de dialogue des paramètres CAN pour maître CAN

Paramètre de base | Paramètre CAN

Taux de bauds: 125000

Com. Cycle Period (µsec): 0

Sync. Window Lenght (µsec): 0

Sync. COB-ID: 128 activer: ☑

Node-Id: 1

☐ Démarrer automatiquement:

☐ DSP301,V4.01 und DSP306 unterstützen

0

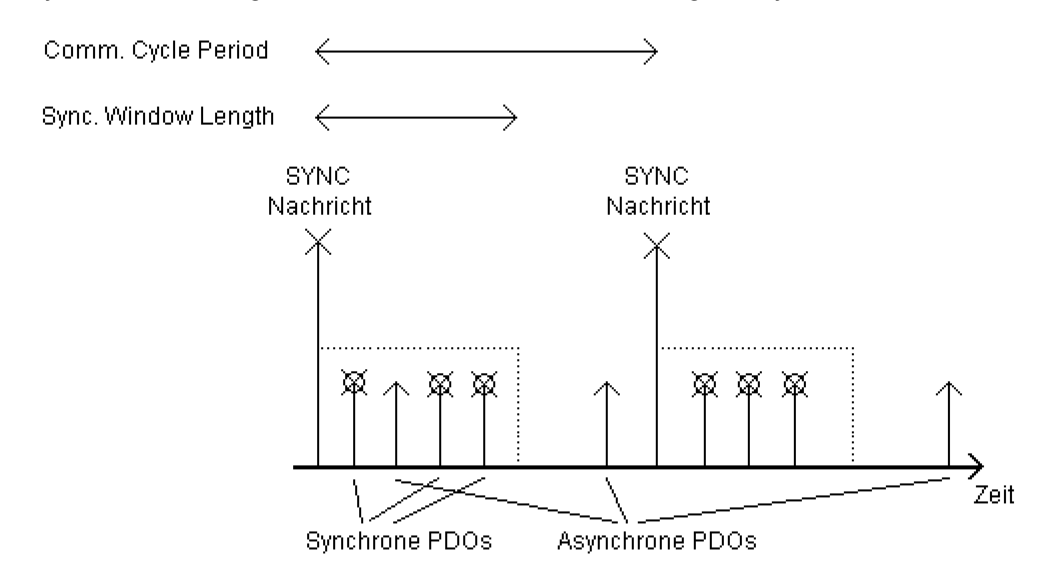
Vous pouvez définir ici les réglages globaux et les paramètres de surveillance pour le Bus CAN :

Configurez le **Débit en bauds** qui doit être utilisé pour la transmission, au moyen de l'option correspondante.

En ce qui concerne les PDO (Process Data Object), on distingue le mode de transmission synchrone du mode de transmission asynchrone (voir les propriétés PDO). Le message de synchronisation est envoyé avec un numéro univoque **Sync.COB-Id** (Communication Object Identifier), dans l'intervalle de temps spécifié dans la zone **Com.**munication **Cycle Period**, en microsecondes.

Activer : c'est uniquement lorsque cette option est activée qu'un envoi de messages de synchronisation a lieu entre le maître et les esclaves.

Les PDO synchrones sont transmis immédiatement après le message de synchronisation dans la fenêtre de temps (**Sync. Window Length** en microsecondes). Si les champs Comm. Cycle Period et Sync. Window Length sont définis avec 0, aucun message de synchronisation n'est envoyé.



Le **Node-Id** sert à identifier de façon univoque le module CAN. Il correspond au numéro configuré dans le module lui-même et est compris entre 1 et 127. L'Id doit être entré en décimal. (À ne pas confondre avec le 'Numéro de nœud' qui est en outre utilisé dans la configuration de l'automate !)

Si l'option **Démarrer automatiquement** est activée, alors le bus CAN est initialisé et lancé automatiquement lors du téléchargement et de l'amorçage de l'automate programmable. Dans le cas contraire, le bus CAN doit être lancé à l'intérieur du projet.

Si l'option **Supporter DSP301,V4.01 et DSP306** est activée, des esclaves modulaires ainsi que des extensions supplémentaires relatives aux normes DSP301 V3.01 et DSP306 seront supportées. Dans ce cas, la cadence pour la fonction d'espionnage Heartbeat peut par exemple être réglée (**Heartbeat Maître [ms]:**), cette fonction pouvant être utilisée comme alternative à la fonction Nodeguarding. Contrairement à cette dernière, les Heartbeats peuvent être envoyés à partir de tous les modules CAN indépendamment les uns des autres. Normalement, on procède à une configuration telle que le maître envoie les Heartbeats aux esclaves.

Paramètre de module d'un CAN-Maître

La boîte de dialogue des paramètres d'une module CAN correspond à celle des autres modules: Les paramètres qui ont été attribués au maître en plus des paramètres CAN dans le fichier de configuration sont représentés ici, et leurs valeurs peuvent normalement être éditées.

Paramètres de base d'un module CAN

La boîte de dialogue des paramètres de base d'un CAN-esclave correspond à celle des autres modules (voir chapitre 6.6.5, Paramètres de base d'un module E/S).

Dans **Adresse d'entrée** et **Adresse de sortie**, on entre, pour une configuration CAN, les adresses CEI qui permettent d'accéder aux PDO (Process Data Object) dans le projet. Le sens (entrée ou sortie) est déterminé par rapport au module.

Paramètre CAN d'un module CAN

Les paramètres CAN d'un module CAN ne reprenant pas la fonction globale de surveillance en tant que 'maître' se différencient de ceux d'un CAN-Maître.

Section Général :

Le **ID de nœud** sert à identifier de façon univoque le module CAN. Il correspond au numéro configuré dans le module lui-même et est compris entre 1 et 127. L'Id doit être entré en décimal.

Si **Ecrire DCF** est activé, un fichier DCF est créé après qu'un fichier EDS ait été inséré dans le répertoire configuré et réservé aux fichiers de compilation. Le nom de ce fichier reprend le nom du fichier EDS, derrière lequel on accole le Node-Id.

Si l'option **Créer tous les SDO** est activée, des SDO sont créés pour tous les objets, pas uniquement pour les objets modifiés.

Si l'option **Remettre le nœud à zéro** est activée, l'esclave procède à une remise à zéro dès que la configuration est chargée sur l'automate.

Section Nodeguard : (alternative à la fonction Heartbeat)

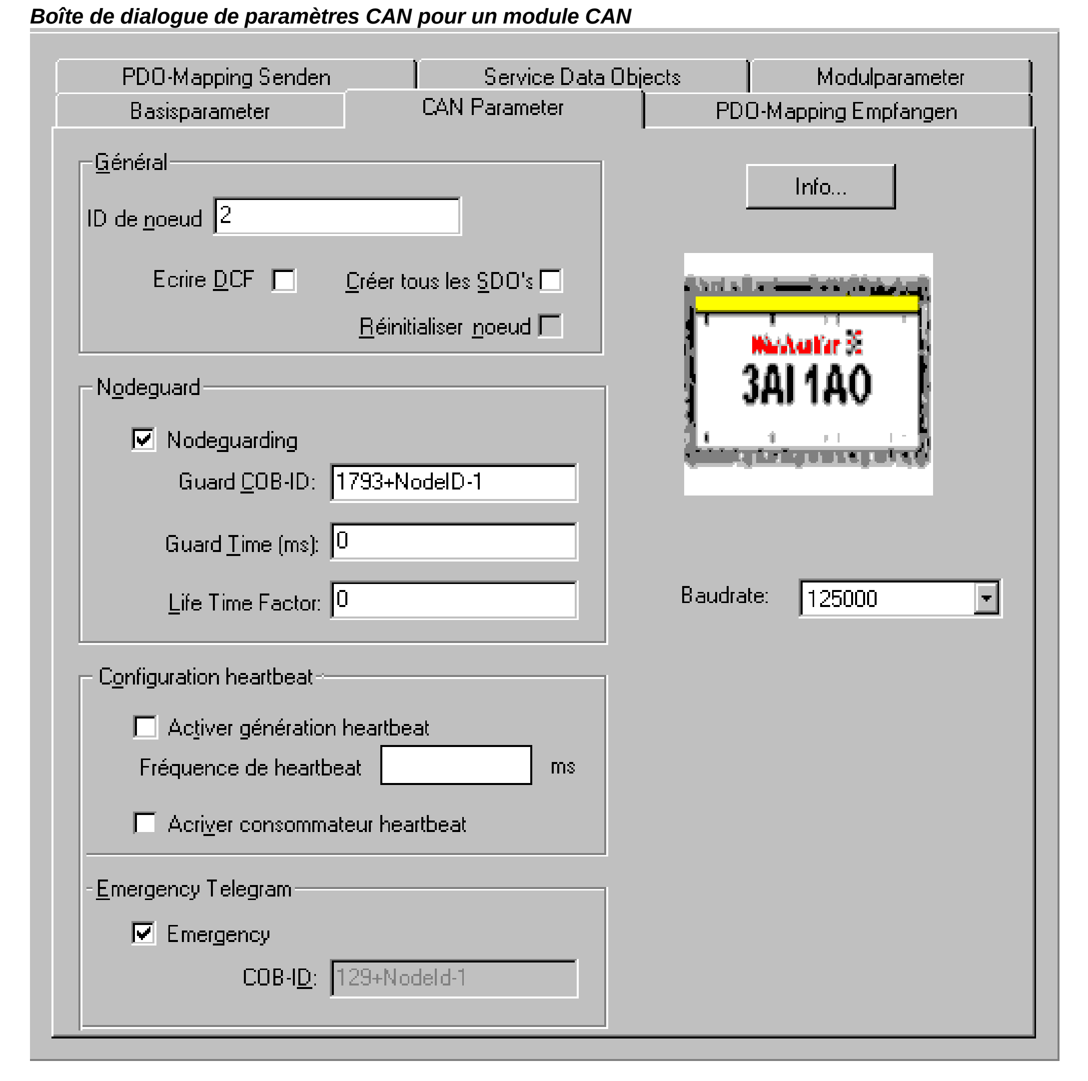
Si l'option **Nodeguarding** est activée, alors un message est envoyé au module dans l'intervalle de temps spécifié au niveau de **Guard Time**, en millisecondes. Si le module ne répond pas par le **Guard COB-ID** (Communication Object Identifier) spécifié, alors l'état Timeout est attribué au module. Si le nombre de tentatives (**Life Time Factor**) est épuisé, alors le module est jugé invalide. L'état du module est affiché au niveau de l'adresse de diagnose (voir ci-dessus). Si rien n'est spécifié dans Guard Time et dans Life Time Factor (0), il n'existe pas de contrôle du module.

Section Configuration Heartbeat : (alternative à la fonction Nodeguard)

Si l'option **Activer génération heartbeat** est activée, le module envoie les Heartbeats selon les intervalles indiqués sous **Heartbeat Producer Time:** en **ms**.

Si l'option **Activer consommateur heartbeat** est activée, le module obéit aux Heartbeats qui sont envoyés par le maître. Dès que de tels Heartbeats ne sont plus reçus, le module met les E/S hors circuit.

Boîte de dialogue de paramètres CAN pour un module CAN



Section Emergency Telegram :

Un module envoie des messages **Emergency** avec un **COB-Id** univoque dans le cas où des erreurs internes surviennent. Ces messages, propres à chaque module, sont affichés au niveau de l'adresse de diagnose.

Derrière le bouton **Info** se cachent les données "FileInfo" et "DeviceInfo" du fichier EDS ou DCF du fabricant du module donné.

Choix de modules CAN pour esclaves modulaires

Les **modules disponibles** pour l'esclave modulaire sont énumérés dans la colonne de gauche. Composez votre propre sélection à l'aide des boutons **Ajouter** et **Effacer** ; cette sélection est affichée dans la colonne de droite (**Modules sélectionnés**). La sélection PDO et SDO est modifiée en conséquence.

Envoyer mappage PDO d'un module CAN

Les onglets **Recevoir mappage PDO** et **Envoyer mappage PDO** de la boîte de dialogue Configuration d'un module CAN permettent de modifier le "mappage" du module défini dans le fichier EDS.

Boîte de dialogue pour l'envoi de mappage PDO

Paramètre CAN | Recevoir mapping PDO | Envoyer mapping PDO | Service Data Objects | P

WriteAnalogueOutput16
StandardDataTypes

>>

Propriétés
Ajouter PDO
Supprimer

PDO 0x1400 (Id: 0x80000000|513+N
PDO 0x1401 (Id: 769+NodeId-1)
PDO 0x1402 (Id: 0x80000000)
PDO 0x1403 (Id: 0x80000000)

Sur le côté gauche se trouvent tous les objets "mappables" du fichier EDS, qui peuvent être ajoutés aux PDO (Process Data Object) situés du côté droit (bouton **>>**) et être à nouveau supprimés (bouton **Supprimer**). Les StandardDataTypes peuvent être insérés afin de créer des espaces vides à l'intérieur du PDO.

Le bouton **Insérer PDO** permet de créer des PDO supplémentaires et de leur affecter des objets appropriés. L'affectation des entrées et des sorties aux adresses CEI s'effectue par l'intermédiaire des PDO insérés. Les configurations effectuées sont affichées dans la configuration de l'automate une fois que l'on a quitté la boîte de dialogue. A cet endroit, des noms symboliques peuvent être attribués aux différents objets.

Les propriétés des PDO, définies dans la norme, peuvent être éditées au moyen de **Propriétés**.

Propriétés PDO Dialogue

Propriétés PDO - 0x1800

COB-ID: 0x80000000|385+NodeId-1
Inhibit Time(100µs): 0
Groupe de priorité CMS: 3
Type de transmission: asynchrone - spécifique au profile de l'
Numéro Sync's:
Temps de l'événement: 0 ms

OK
Annuler

Chaque message PDO nécessite un **COB-Id** (Communication Object Identifier) univoque.

Si une des options du module n'est pas supportée ou s'il n'est pas permis de modifier la valeur, alors le champ apparaît en gris et il n'est pas possible de l'éditer.

Inhibit Time est le temps minimum entre deux messages d'un PDO donné. Cela permet d'empêcher que des PDO transmis après la modification de la valeur ne soient envoyés trop souvent.

Le **Groupe de priorité CMS** ne peut pas être modifié. Il définit la priorité relative des PDO pour la transmission CAN. Les numéros affichés vont de 0 à 7, le chiffre 0 correspondant à la priorité maximale.

Type de transmission vous permet d'accéder à une sélection des différents modes de transmission pour un module donné:

acyclique - synchrone: le PDO est transmis en mode synchrone mais non périodique.

cyclique – synchrone: le PDO est transmis en mode synchrone et **Numéro Sync's** indique le nombre de messages de synchronisation existant entre deux transmissions, pour ce même PDO.

synchrone – RTR seulement: le PDO est actualisé après un message de synchronisation, mais n'est pas envoyé. Il n'est transmis qu'à la suite d'une demande explicite (**R**emote **T**ransmission **R**equest).

asynchrone – RTR seulement: le PDO est actualisé et transmis uniquement à la suite d'une demande explicite (**R**emote **T**ransmission **R**equest).

asynchrone – spécifique au profile de l'appareil et **asynchrone - spécifique au fabricant: le PDO** est transmis uniquement après certains événements.

Numéro Sync's : selon le type de transmission, ce champ est éditable pour permettre la saisie du nombre de messages de synchronisation (définition dans la boîte de dialogue des paramètres CAN, Com. Cycle Period, Sync Window Length, Sync. COB-Id) après lequel le PDO doit à nouveau être envoyé.

Event-Time : selon le type de transmission, l'intervalle de temps entre deux transmissions de PDO est indiqué ici en millisecondes (**ms**).

Service Data Objects

Tous les objets du fichier EDS ou DCF compris dans la zone allant de Index 0x2000 à 0x9FFF et identifiés comme des objets définissables sont énumérés ici.

Boîte de dialogue pour la configuration des Service Data Objects (SDO)

Recevoir mapping PDO | Envoyer mapping PDO | Service Data Objects | Paramètre de modu

Indexe	Nom	Valeur	Type	Préférence
6411su...	Output_1H		Integer16	
6420su...	Input_1H		Unsigned16	
6420su...	Input_2H		Unsigned16	
6420su...	Input_3H		Unsigned16	
6421su...	Input_1H		Unsigned8	
6421su...	Input_2H		Unsigned8	
6421su...	Input_3H		Unsigned8	
6423	GlobalEnableInterrupts	0	Boolean	0
6424su...	Input_1H		Integer32	

Un **Index**, un **Nom**, une **Valeur**, un **Type** et un **Préférence** sont spécifiés pour chaque objet. La valeur peut être modifiée. Sélectionnez la valeur de votre choix et appuyez sur la <Barre d'espacement>. Après que la modification ait été effectuée, vous pouvez confirmer la valeur au moyen de la touche <Entrée> ou la rejeter au moyen de la touche <Echap>.

Les valeurs configurées sont transmises aux modules CAN sous forme de SDO (Service Data Object), au moment de l'initialisation du bus CAN.

Remarque : Tout type de données pour lequel il n'existe pas de compatibilité entre CANopen et la norme CEI-61131, est remplacé dans CoDeSys par le type de données CEI-61131 qui est le plus proche et de taille plus grande que le type d'origine.

6.6.10 Configuration CANDevice (CANopen-Esclave)

Un automate programmable CoDeSys peut apparaître dans un réseau CAN sous la forme d'un esclave CANopen (également appelé Nœud CANopen et désigné dans la suite par CanDevice).

Pour ce faire, l'automate peut être configuré dans le configurateur d'automate CoDeSys, et la configuration peut être sauvegardée comme fichier d'appareil (fichier EDS). Ce fichier EDS peut alors être utilisé au gré dans une configuration de Maître CANopen.

Conditions pour la création d'un CanDevice dans le configurateur d'automate CoDeSys :

- Les bibliothèques

  - 3S_CanDrv.lib
  - 3S_CanOpenManager.lib
  - 3S_CanOpenDevice.lib

  sont reliées au projet CoDeSys. Elles sont nécessaires afin de pouvoir exploiter un automate comme appareil CANopen.

- Dans le fichier de configuration (*.cfg) servant de base, il y a déjà une entrée appropriée pour un CanDevice, si bien que l'on puisse insérer un sous-élément 'CanDevice' dans le configurateur d'automate et le paramétrer dans les trois dialogues de configuration ci-dessous:

  - Configuration de base
  - Configuration CAN
  - Default PDO-Mapping

Configuration de base, CANDevice

Boîte de dialogue pour Configuration de base

Configuration de l'automate
Configuration de
CanDe[V
Configuration de base | Configuration CAN | Défaut mapping PDO
Nom du bus: CAN1
Nom de la tâche mise à jour: t1
Génération du fichier EDS
Générer fichier EDS
Nom du fichier EDS:
D:\proj1\.cconf\candev.EDS
Balayer...
Défaut pour fichier EDS:
D:\proj1\cconf\templ\candev.tpl
Balayer...

Nom de bus: actuellement inutilisé.

Nom de la tâche mise à jour: nom de la tâche dans laquelle a lieu l'appel du CanDevice.

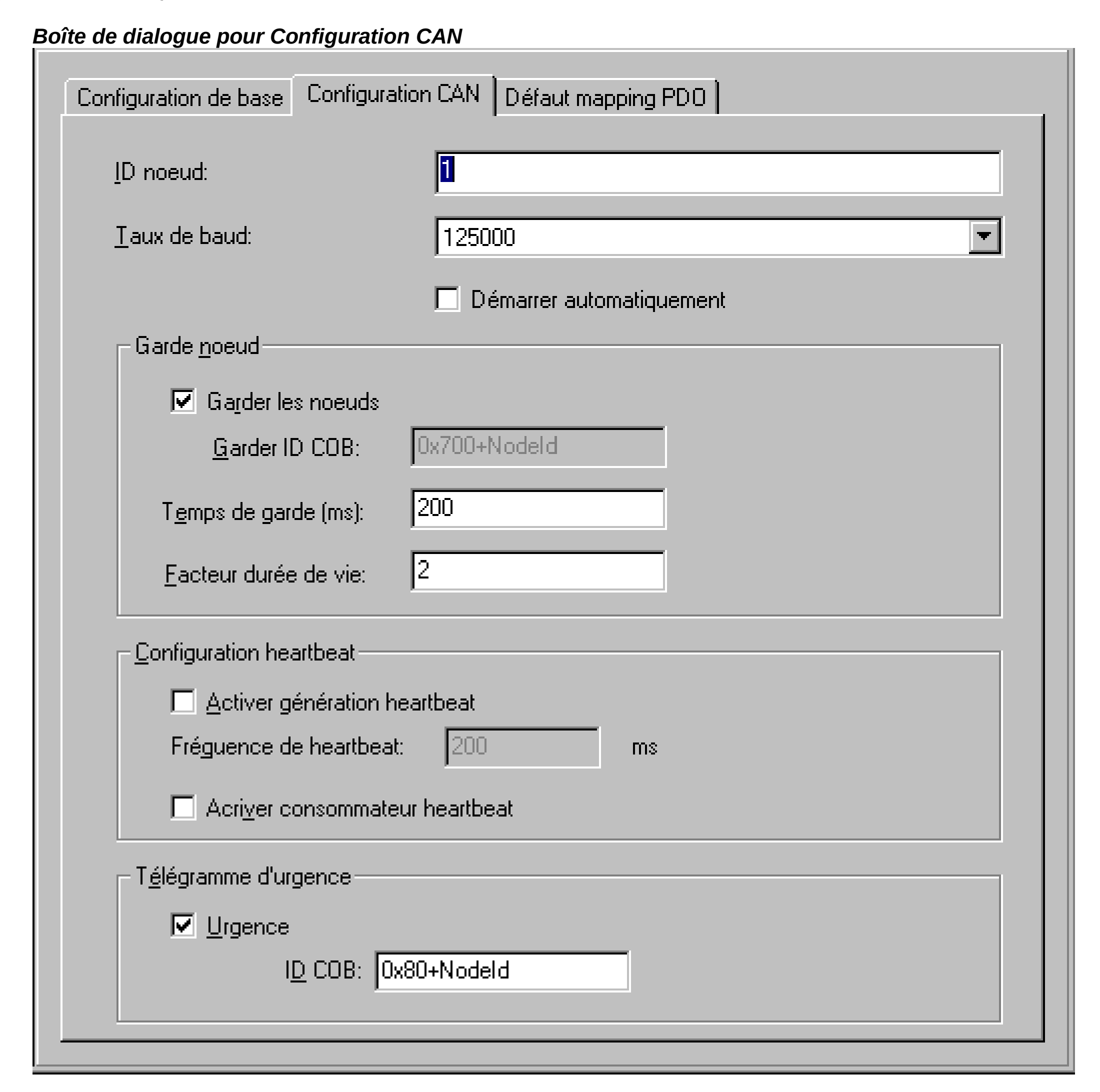
Générer fichier EDS: si un fichier EDS a été créé ici à partir des configurations, cela afin de pouvoir insérer le CanDevice dans n'importe quelle configuration maître, cette option doit être activée et le nom du fichier EDS doit être indiqué.

En option, on peut également indiquer un fichier modèle créé manuellement (**Défaut pour fichier EDS**) auquel la configuration du CanDevice peut être ajoutée. À titre d'exemple, vous pouvez sauvegarder certaines entrées qui seraient pertinentes pour plusieurs fichiers EDS dans un fichier

texte "EDS_template.txt" et indiquer ce fichier à cet emplacement. Lorsqu'un fichier EDS "device_xy.eds" est alors créé pour la configuration CanDevice à partir du projet actuel, les entrées dans ce fichier sont sauvegardées en même temps que celles de "EDS_template.txt" dans "device_xy.eds". (Veillez à ne pas utiliser l'extension ".eds" pour le modèle!) Lorsque des entrées déjà définies dans le modèle sont créées à partir du projet, les données du modèle ne sont pas écrasées.

Configuration CAN, CANDevice

Boîte de dialogue pour Configuration CAN

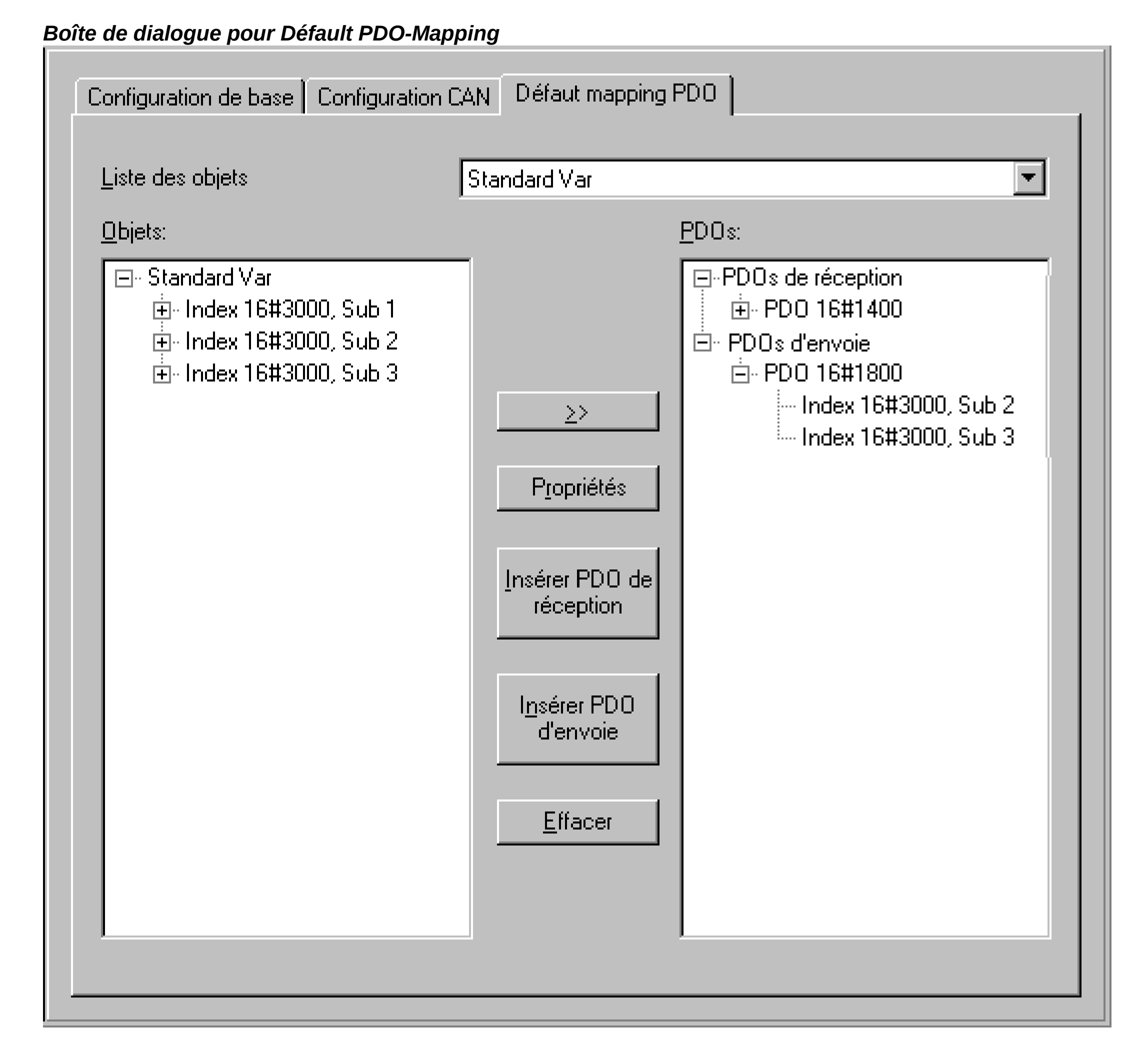


Vous pouvez configurer ici l'**ID noeud'** ainsi que le **Taux de baud**. L'ID nœud est le numéro du nœud grâce auquel le maître peut s'adresser à l'appareil dans le réseau CANopen.

Il est possible de configurer les fonctions **Garde-ID COB** et le **Télégramme d'urgence**. Reportez-vous pour ce faire aux descriptions ad hoc pour la configuration des autres modules CAN et maîtres. (Heartbeat n'est actuellement pas encore supporté.)

Défault PDO-Mapping, CANDevice

Boîte de dialogue pour Défault PDO-Mapping



On peut définir ici l'attribution des entrées dans un répertoire local d'objets (Gestionnaire des paramètres) aux PDO qui sont émises / reçues par le CanDevice. Les PDO sont alors à disposition lors de l'utilisation du CanDevice dans une configuration maître pour le mappage PDO local.

Dans le gestionnaire des paramètres, la liaison avec les variables de l'application est établie dans les entrées (répertoire de variables) par le biais de l'index/du sous-index. Notez que le sous-index 0 d'un index comprenant plus d'un sous-index est implicitement utilisé pour la mention du nombre de sous-index. Les objets (paramètres) d'un index doivent être saisis par ordre croissant (après le sous-index) dans une liste de paramètres.

Liste des objets mappables: Sélectionnez dans la liste des paramètres la liste de variables pour les entrées desquelles le CanDevice doit créer des PDO.

En fonction de la liste sélectionnée, les objets apparaissent dans la fenêtre de sélection gauche. La configuration des PDO est créée dans la fenêtre de droite (PDO's). Grâce aux boutons **Insérer PDO de réception** et **Insérer PDO d'envoi**, vous pouvez créer en dessous des éléments de dossier '**PDO de réception**' et '**PDO d'envoi**' des PDO servant à l'envoi ou à la réception. Afin d'attribuer à un des PDO d'envoi / de réception un objet de la liste de gauche, marquez cet objet dans la fenêtre de gauche ainsi que le PDO dans la fenêtre de droite, puis appuyez sur le bouton >>. L'objet est alors rattaché en dessous du PDO dans la fenêtre de droite. Les **Propriétés** du PDO peuvent être définies par le biais du dialogue obtenu à partir de la configuration des modules CAN sous un maître.

Le bouton **Effacer** permet d'effacer de la configuration le PDO actuellement marqué dans la fenêtre de droite.

Exemple:

But : La variable PLC_PRG.a doit être reçue sur le premier PDO de réception (COB-Id = 512 + NodeId) du CanDevice.

Un index / sous-index doit donc être relié à la variable PLC_PRG.a dans un répertoire de variables dans le gestionnaire des paramètres: Afin d'ouvrir le gestionnaire des paramètres, ce dernier doit être activé dans la Configuration du système cible sous 'Fonctions de réseau', et il doit être paramétré avec des plages d'index / de sous-index pertinentes.

Ensuite, l'entrée d'index / de sous-index doit être attribuée comme entrée de mappage à un PDO dans la boîte de dialogue pour le mappage PDO par défaut du CanDevice.

6.6.11 La configuration de l'automate en mode en ligne

La configuration de l'automate en mode en ligne

En mode En Ligne, l'éditeur de configuration affiche les états des entrées et sorties de l'automate. Si une entrée ou une sortie de type booléen a la valeur 'TRUE', alors la petite boîte située devant l'entrée ou la sortie est visualisée dans l'arborescence de configuration en bleu, tandis que des valeurs non booléennes sont complétées à la fin de l'entrée (p. ex."=12"). Les entrées de type booléen peuvent être activées/désactivées en cliquant avec la souris, alors que dans le cas des autres entrées une boîte de dialogue s'affiche pour entrer la nouvelle valeur et pour autant que vous ayez clique sur le début de la ligne. La nouvelle valeur est activée dans l'automate programmable dès que la confirmation a été effectuée au moyen de **OK**.

Notez en outre, selon le système cible, les possibilités de Diagnostic En ligne.

6.6.12 Analyse matérielle/Informations/Diagnostic du système cible

Si le système cible et le fichier de configuration utilisé le supportent, vous pouvez appeler, à partir du système cible, des informations relatives à la configuration et au statut actuel ainsi qu'au diagnostic des modules de matériel existants, et utiliser ou afficher ces informations dans la Configuration de l'automate dans CoDeSys.

Numériser la configuration du module

Si le système cible et le fichier de configuration actuel le supportent, la commande **Numériser la configuration du module** est disponible dans le menu contextuel pour le module actuellement marqué dans la Configuration de l'automate. Cette commande n'est disponible qu'en **mode Hors ligne** et fait que la configuration de matériel actuelle de ce module puisse être lue à partir de l'automate programmable et que les sous-nœuds éventuellement disponibles dans l'arborescence de configuration puissent être proposées pour insertion. Si le fichier de configuration le permet, on peut ainsi illustrer d'une manière simple dans CoDeSys la configuration existante du module.

Charger l'état du module

Si le système cible et le fichier de configuration actuel le supportent, la commande **Charger l'état du module** est disponible dans le menu contextuel pour le module actuellement marqué dans la Configuration de l'automate. Cette commande n'est disponible qu'en mode En ligne et fait que le statut actuel du module puisse être lu à partir de l'automate programmable et qu'il soit affiché dans l'arborescence de configuration dans une couleur spécifique :

Noir : le module est disponible et est correctement paramétré.

Bleu : le module est disponible mais n'est pas correctement paramétré.

Rouge : le module n'est pas disponible.

Cette représentation du statut s'effectue également automatiquement à chaque téléchargement.

Afficher les messages de diagnose

Si le système cible et le fichier de configuration actuel le supportent, la commande **Afficher les messages de diagnostic** est disponible dans le menu contextuel pour le module actuellement marqué dans la Configuration de l'automate. Cette commande n'est disponible qu'en mode En ligne et fait que les informations actuelles relatives au diagnostic du module puissent être lues à partir de l'automate programmable et affichées dans une fenêtre.

6.7 Configuration des tâches

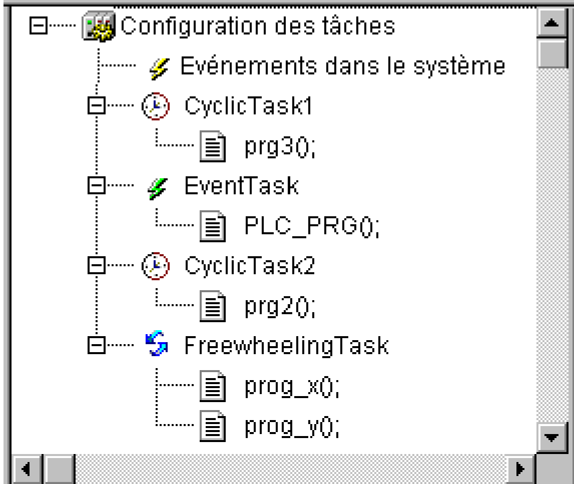
6.7.1 Aperçu de la Configuration des tâches

L'exécution d'un projet peut être commandée non seulement via le programme spécial PLC_PRG, mais également via le gestionnaire des tâches.

- Une **tâche** est une unité d'exécution dans le déroulement d'un programme CEI. Elle est définie par un nom, une priorité et un type, qui détermine la condition de son démarrage. Cette condition est soit définie dans le temps (intervalles cycliques, spontané), soit par un événement interne ou externe qui déclenche la tâche devant être exécutée ; par exemple le front montant d'une variable globale de projet ou un événement d'interruption de l'automate.
- Chaque tâche peut se voir attribuer une suite de programmes qui devront être exécutés lors de l'exécution de la tâche.
- L'interaction des priorités et des conditions détermine l'ordre chronologique selon lequel les tâches seront exécutées.
- Vous pouvez configurer un Contrôle de temps (Watchdog) pour chaque tâche.
- L'exécution de la tâche peut être suivie en mode En ligne dans une représentation graphique.
- Vous pouvez en outre associer directement des événements dans le système (p.ex. Start, Stop, Reset) à l'exécution d'un module de projet.

La **configuration des tâches** est contenue sous forme d'objet dans l'onglet **Ressources** de l'Organisateur d'objets. L'éditeur des tâches se présente sous la forme d'une fenêtre scindée en deux.


Exemple de configuration des tâches



Les tâches sont représentées dans la partie gauche de la fenêtre sous la forme d'une **arborescence de configuration**. La première ligne contient 'Configuration des tâches', ensuite l'entrée 'Événements dans le système' puis les entrées pour chaque tâche individuelle représentées par les noms de tâches. Les appels de programme correspondants sont rattachés en dessous de chaque entrée de tâche.

La partie droite de la fenêtre est réservée au **Dialogue des caractéristiques** de l'entrée marquée dans l'arborescence de configuration. Les tâches individuelles, appels de programme ou événements dans le système peuvent être définis ici. Les possibilités de configuration disponibles dans la boîte de dialogue des caractéristiques sont spécifiques au système cible et sont définies en format XML par le biais d'un **fichier descriptif** référencé dans le fichier cible. Si les définitions standard ont été complétées par des définitions spécifiques au client, celles-ci sont disponibles pour la configuration dans un onglet supplémentaire 'Paramètres' dans la partie droite de la fenêtre.

Remarque : Il est conseillé de ne pas utiliser les mêmes fonctions de chaînes de caractères (voir Bibliothèque standard standard.lib) dans plusieurs tâches car un écrasement pourrait dans ce cas survenir lors de l'exécution des tâches.

6.7.2 Travailler avec le configurateur de tâches

Vous trouvez les commandes les plus importantes dans le menu contextuel (touche droite de la souris).

Les mots "Configuration des tâches" sont écrits dans l'en-tête de la configuration des tâches. Si un signe plus se trouve devant ce mot, alors la liste des entrées des tâches est masquée. En double-cliquant sur la liste ou en appuyant sur la touche <Entrée>, celle-ci est affichée. Un signe moins apparaît alors et il suffit de double-cliquer à nouveau pour masquer la liste.

Une liste d'appels de programmes, pouvant également être affichée ou masquée, est reliée à chaque tâche.

- La commande 'Insérer' 'Insérer une tâche' permet d'insérer une tâche après l'entrée marquée.
- La commande 'Insérer' 'Ajouter sous-élément' permet d'ajouter une tâche à la fin de l'arborescence de configuration.
- La commande 'Insérer' 'Ajouter Appel de programme' permet d'ajouter un appel de programme à une tâche marquée dans l'arborescence de configuration.

La configuration d'une entrée sélectionnée dans l'arborescence de configuration s'effectue dans la boîte de dialogue des caractéristiques située dans la partie droite de la fenêtre, en (dés)activant les options ou par des entrées dans les champs de saisie. Vous avez alors accès à la boîte de dialogue permettant la définition des caractéristiques des tâches (voir 'Insérer tâche'), la boîte de dialogue permettant l'ajout d'un appel de programme (voir 'Insérer' 'Ajouter Appel de programme'), ou encore le tableau des Événements dans le système. Les réglages entrepris seront immédiatement repris dans l'arborescence de configuration et y seront affichés dès que celui-ci sera à nouveau actif (à l'avant-plan).

Un nom de tâche ou de programme peut également être directement édité dans l'arborescence de configuration. Pour ce faire, il faut cliquer sur le nom ou appuyer sur la barre d'<Espace> en ayant préalablement marqué l'entrée ; une possibilité d'édition s'ouvre alors dans laquelle vous pouvez procéder à la modification du nom.

Les touches directionnelles vous permettent de marquer dans l'arborescence de configuration l'entrée précédente ou suivante.

'Insérer Tâche' ou 'Ajouter sous-élément'

Cette commande vous permet d'ajouter une nouvelle tâche au niveau de la configuration des tâches. Les entrées se composent toutes d'un symbole et d'un nom de tâche.

Si vous sélectionnez dans l'arborescence de configuration une entrée de tâche ou encore l'entrée **Événements dans le système**, la commande 'Insérer tâche' est disponible. La nouvelle tâche est insérée après la tâche sélectionnée. Si "Configuration des tâches" est sélectionnée, alors la commande "**Ajouter tâche**" est disponible et toute nouvelle tâche est insérée à la fin de la liste existante.

Le nombre maximal de tâches est défini par le système cible. Notez qu'un certain nombre de tâches est déjà réservé le cas échéant par la configuration de l'automate, pour certains modules (définition dans le fichier .cfg actuel).

Lorsqu'une tâche est insérée, la boîte de dialogue pour définir les attributs des tâches est affichée.

Boîte de dialogue pour la définition des attributs de tâches

Configuration des tâches
Evénements dans le
CyclicTask1
prg2();
EventTask
PLC_PRG();
NeueTask
FreewheelingTask
prog_x();
prog_y();

Attributs des tâches
Nom: NeueTask
Priorité: (0-31): 4
Type
Cyclique
A répétion auto
Déclenché par événements
Déclenché par événements
Propriétés
Evénement: t#100ms ms
Surveillance
Surveillance
Temps (p.ex. t#200ms): t#150ms ms
Sensibilité: 1

Entrez les attributs souhaités :

Nom : il s'agit du nom de la tâche, il sera également affiché dans l'arborescence de configuration ; ce nom pourra aussi être édité en cet endroit en cliquant dessus ou en appuyant sur la barre d'<Espace>, ce qui aura pour effet d'ouvrir un champ de saisie.

Priorité (0 - 31) : un chiffre définissant la priorité, 0 étant le plus prioritaire et 31 le moins prioritaire.

Type:

Cyclique ( ) : la tâche est démarrée par cycles conformément au temps entré au champ Intervalle.

A repetion auto ( ) : la tâche se déclenche au démarrage du programme et est redémarrée à la fin de chaque exécution. On ne donne pas de temps de cycle.

Déclenché par événements ( ) : la tâche est démarrée dès que la variable associée à l'événement obtient un front montant.

Déclenché par événement externe ( ) : la tâche est démarrée dès que l'événement du système associé à l'événement intervient. Les événements supportés présentés dans la liste de sélection sont spécifiques au système cible et sont également définis via le fichier cible.

Propriétés :

Intervalle (pour le type 'cyclique' ou 'spontané') : l'intervalle de temps après lequel la tâche doit redémarrer. Si un nombre est saisi, l'unité - milliseconde [ms] ou microseconde [µs] - peut être choisie dans le champ de sélection suivant le nombre. Les entrées en millisecondes apparaissent en format TIME (p. ex. "t#200ms") dès que l'on passe à autre chose; vous pouvez également écrire cet intervalle sur la ligne de saisie directement dans ce format. Pour les entrées en microsecondes, le chiffre est seul représenté (p. ex. "300").

(pour le type 'Déclenché par événements' ou 'Déclenché par événement externe') : une variable globale qui doit procéder à l'exécution de la tâche après un front montant. Le bouton ou la touche vous permet d'accéder à la liste de sélection pour l'édition en vue de choisir parmi les variables globales disponibles.

Si vous ne procédez à aucune entrée dans les deux champs, l'intervalle d'exécution dépend du système d'exécution utilisé (reportez-vous à cet effet à la documentation spécifique du système d'exécution ; par exemple, un intervalle de 10 ms est inséré dans CoDeSys SP NT à partir de la version V2.2).

Surveillance

Activer surveillance Activez cette option (☑) si vous souhaitez que la tâche se termine avec un statut d'erreur dès que l'exécution de celle-ci dépasse le temps de surveillance alloué sous la rubrique 'Temps' (mécanisme de surveillance (Watchdog).

Temps (p. ex. : t#200ms) : temps de surveillance; après ce temps, le mécanisme de surveillance est activé à moins que la tâche ne se soit interrompue d'elle-même.

Sensibilité : nombre de dépassements du temps de surveillance pouvant être accepté sans que l'automate ne se mette en mode d'erreur.

'Insérer' 'Ajouter appel de programme' ou 'Insérer' 'Insérer appel de programme

Ces commandes vous permettent d'ouvrir la boîte de dialogue pour entrer un appel de programme relatif à une tâche, à l'intérieur de la configuration des tâches. L'entrée dans l'arborescence de configuration se compose d'un symbole (📄) et du nom du programme.

La commande "**Insérer appel de programme**" entraîne l'insertion du nouvel appel de programme devant l'appel de programme sélectionné, alors que la commande "**Ajouter appel de programme**" permet d'insérer l'appel de programme à la fin de la liste existante des entrées de programmes.

Boîte de dialogue pour l'entrée d'un appel de programme



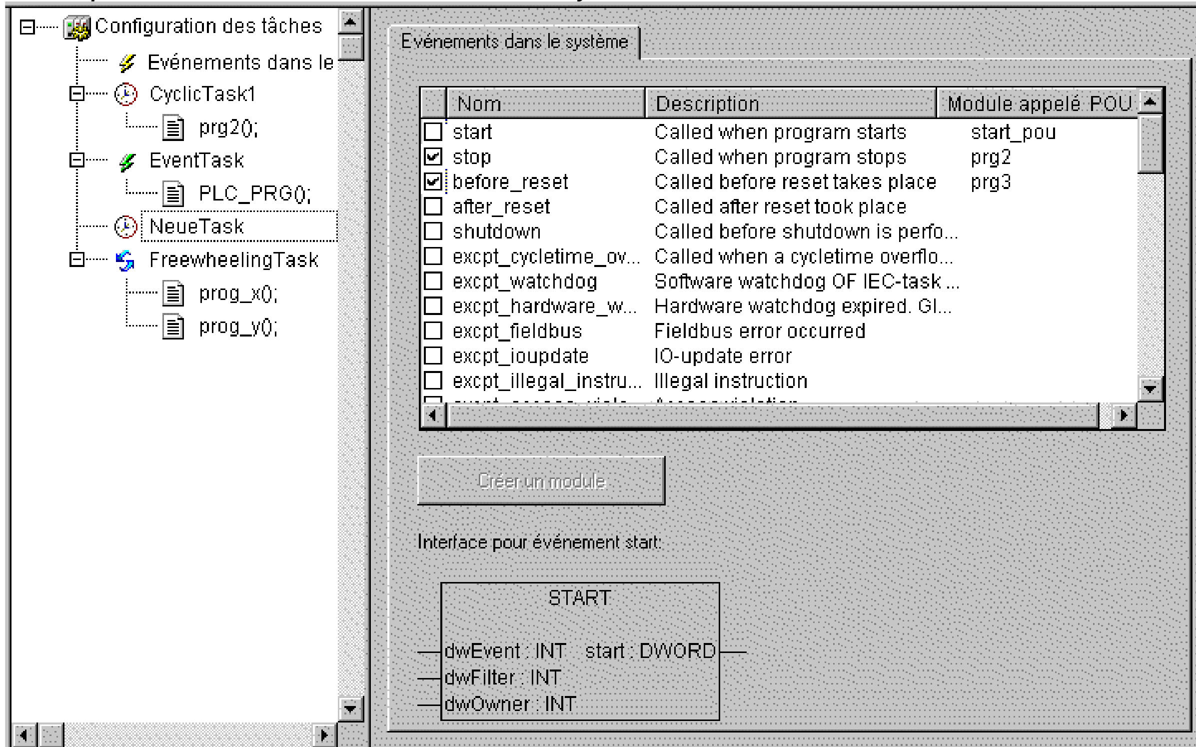
Saisissez dans le champ Appel de programme un nom de programme valide dans votre projet ou bien ouvrez la liste de sélection pour l'édition au moyen du bouton **...** ou de la touche <F2> pour choisir parmi les noms de programme valides. Le nom du programme peut encore être modifié dans l'arborescence de configuration pour autant que l'entrée de programme ait été sélectionnée. Pour ce faire, ouvrez un champ d'édition en cliquant sur le nom ou en appuyant sur la barre <Espace>. Si le programme sélectionné nécessite des variables d'entrée, spécifiez-les de la manière habituelle, en utilisant le type déclaré (p.ex. . prg(invar:=17)).

6.7.3 Événements dans le système

Outre une tâche, un événement du système (Event) peut également appeler l'exécution d'un module du projet. Les événement du système utilisables à cet effet dépendent du système cible (définition en format XML par le biais d'un fichier descriptif référencé dans le fichier cible). Ils se composent de la liste des événements standard de système relatifs à l'automate et éventuellement des événements supplémentaires spécifiques au fabricant. Des événements possibles sont par exemple Stop, Start, Changement En ligne.

L'assignation des événements de système à chaque module à appeler s'effectue par le biais de la boîte de dialogue **Événements** qui s'ouvre dès que l'entrée "⚡ Evènements dans le système" est marquée dans l'arborescence de configuration.

Tableau pour la définition des tâches d'événements du système



Chaque événement est représenté sur une ligne du tableau : **Nom** et **Description** sont repris hors de la description du système cible, et vous pouvez insérer dans la colonne **Module appelé POU** le module du projet qui doit être exécuté lors de l'intervention de l'événement.

Pour ce faire, vous pouvez utiliser la liste de sélection pour l'édition (<F2>) ou saisir manuellement le nom d'un module existant (par exemple "PLC_PRG" ou "PRG.ACT1") ou encore le nom d'un module qui n'est pas encore disponible. De façon à créer ce dernier au sein du projet, appuyez sur le bouton **Créer un module**. Le module apparaît alors dans l'Organisateur d'objets et contient automatiquement dans la partie Déclaration les définitions des paramètres éventuellement nécessités pour l'événement.

Ce paramétrage d'un événement nécessité selon les circonstances est également représenté graphiquement en tant que module en dessous de la liste d'attribution, et ce dès que l'entrée correspondante est marquée dans le tableau.

L'appel d'un module par l'événement ne se produit que lorsque l'entrée est activée, à savoir lorsque la case dans la première colonne est munie d'un crochet (☑). L'**activation** ou la **désactivation** s'effectue par le biais d'un clic sur cette case.

6.7.4 Configuration des tâches en mode En Ligne

En mode En ligne, le statut et le nombre de cycles exécutés pour chaque tâche est indiqué dans l'arborescence de configuration, et le comportement dans le temps peut être suivi par le biais d'une représentation graphique. Il faut pour ce faire que les bibliothèques **SysTaskInfo.lib** et **SysTime.lib** soient associées au projet. Les fonctions de bibliothèques sont utilisées de manière interne pour évaluer l'exécution des tâches.

<u>Affichage du statut dans l'arborescence de configuration :</u>

Pour chaque tâche, le statut actuel ainsi que le nombre de cycles déjà exécutés sont repris en mode En ligne entre les symboles < et > après l'entrée de la tâche : le cycle d'actualisation pour cet affichage correspond à celui de tout autre espionnage. Voici les états possibles :

Idle pas exécuté depuis la dernière mise à jour, ceci vaut principalement pour des tâches événementielles

Running exécuté au moins une fois depuis la dernière mise à jour

Stop	arrêté
Stop on BP	arrêté, point d'arrêt atteint dans cette tâche
Stop on Error	erreur, par exemple division par zéro, erreur de pagination, etc.
Stop Watchdog	dépassement du temps de cycle

L'entrée de la tâche vire au rouge dans les cas 'Stop on Error' et 'Stop Watchdog'.

Représentation graphique de l'évolution dans le temps de la tâche

La charge est représentée pour toutes les tâches sous la forme d'un graphique en histogramme dans la fenêtre de droite dès que l'entrée 'Configuration des tâches' est marquée dans l'arborescence de configuration.

Affichage de la charge de tâche en mode En ligne

Configuration des tâches
Configuration des tâches
Evénements dans le système
CyclicTask1 [RUN, 61, DEBUG]
prg3();
EventTask [RUN, 2]
PLC_PRG();
CyclicTask2 [RUN, 309]
prg2();
FreewheelingTask [RUN, 78]
prog_x();
prog_y();
Exécution des tâches
Déverrouiller
Graduation (µs/pixel): 0.1
CyclicTask1
Min: 9µss Akt: 11µs Max: 12µs Zyklus: 250ms Jitter: 12msµs
EventTask
Min: 10µs Akt: 10µs Max: 11µs Zyklus: 0µs Jitter: 5231ms
CyclicTask2
Min: 9µss Akt: 11µs Max: 13µs Zyklus: 50ms Jitter: 11msµs
FreewheelingT
Min: 9µs Akt: 10µs Max: 11µsµs Zyklus: 200ms Jitter: 12msµs

Chaque tâche est représentée par une barre, dont la longueur indique la durée du cycle. Les mesures suivantes sont indiquées de gauche à droite en dessous de et également dans la barre par des marquages adéquats :

Min :	temps d'exécution minimal mesuré en µs
Akt :	temps d'exécution mesuré en dernier lieu en µs
Max :	temps d'exécution maximal mesuré en µs
Zyklus :	Durée totale du cycle en µs
Jitter :	gigue maximale mesurée en µs

Le bouton **Remettre à zéro** permet de remettre à zéro les valeurs Min. , Max. et Gigue.

L'échelle de la représentation graphique (microsecondes par pixel) peut être réglée via une liste de sélection **Échelle [µs/Pixel]**.

Fonctions En ligne supplémentaires dans le menu contextuel ou dans le menu 'Extras' :

- 'Définir tâche de débogage'
- 'Afficher hiérarchie d'appel'

Quelle tâche est traitée?

Les règles d’exécution suivantes s’appliquent:

- Toute tâche est exécutée si sa condition est vérifiée, c.-à-d. dès que le temps spécifié pour l’intervalle est écoulé ou après un front montant de la variable conditionnelle associée à l’événement.
- Si la condition d’exécution de plusieurs tâches est vérifiée, alors la tâche possédant la plus haute priorité est exécutée.
- Si les conditions d’exécution de plusieurs tâches de même priorité sont vérifiées, alors la tâche ayant le délai d’attente le plus long est exécutée.

'Extras' 'Définir tâche de débogage'

Dans le cas des systèmes cible avec "environnement multitâches préemptif", cette commande permet, en mode En ligne, de spécifier une tâche dans la configuration des tâches; cette tâche sera alors déboguée. Le texte "[DEBUG]" apparaît alors derrière l'entrée de la tâche dans l'arborescence de configuration. Les fonctions de débogage ne s'appliquent qu'à cette tâche, autrement dit, le programme ne s'arrête à un point d'arrêt que si le programme est parcouru par la tâche spécifiée.

La définition de la tâche de débogage est enregistrée dans le projet et est automatiquement reprise lors d'une ouverture de session / lors d'un téléchargement.

Extras' 'Activer/Désactiver tâche'

Cette commande permet de (dés)activer la tâche qui est marquée dans la configuration des tâches. Une tâche désactivée n'est pas prise en compte dans l'exécution du programme. Cette même tâche inactive est représentée en gris clair dans l'arborescence de configuration.

'Extras' 'Afficher hiérarchie d'appel'

Lorsqu’un point d’arrêt est atteint durant un débogage, la hiérarchie d’appel du module concerné peut être déterminée par le biais de cette commande. Il faut pour ce faire sélectionner la tâche de débogage dans l’arborescence de configuration de la configuration des tâches. La fenêtre **Hiérarchie d’appel de la tâche <nom de la tâche>** s’ouvre affichant le module dans lequel un point d’arrêt a été défini (p.ex. "prog_x (2)" pour la ligne 2 du module prog_x). Viennent ensuite les entrées des positions de module appelantes, en ordre inverse. Si vous appuyez sur le bouton **Aller vers**, on accède à la position marquée.

6.8 Gestionnaire d'espion et de recettes

6.8.1 Aperçu de la Gestionnaire d'espion et de recettes

Le gestionnaire d'espion et de recettes permet d'afficher les valeurs des variables choisies. Le gestionnaire d'espion et de recettes permet aussi de préaffecter des valeurs déterminées à des variables et de les transférer en une fois vers l'automate programmable ('Ecrire la recette'). De même, des valeurs actuelles de l'automate programmable peuvent être récupérées et enregistrées comme préaffectations dans le gestionnaire d'espion et de recettes ('Lire la recette'). Ces fonctions sont par exemple utiles pour configurer et enregistrer les paramètres de réglage.

Toutes les listes d'espion créées ('Insérer' 'Nouvelle liste d'espion') sont affichées dans la colonne de gauche du gestionnaire d'espion et de recettes et peuvent être sélectionnées en cliquant avec la souris ou au moyen des touches directionnelles. Dans la partie droite du gestionnaire d'espion et de recettes sont affichées les variables correspondantes.

Pour travailler avec le gestionnaire d'espion et de recettes, ouvrez l'objet Gestionnaire d'espion et de recettes dans l'onglet **Ressources** de l'Organisateur d'objets.

En *mode hors ligne*, il est possible de créer dans le gestionnaire d'espion et de recettes plusieurs listes d'espion en utilisant la commande 'Insérer' 'Nouvelle liste d'espion'.

Pour entrer les variables que l'on souhaite observer, il est possible d'appeler une liste de toutes les variables au moyen de la Liste de sélection pour l'édition. On peut aussi entrer les variables via le clavier, en respectant la formulation suivante:

<Nom de module>.<Nom de variable>

Pour les variables globales, le nom du module n'est pas précisé. Elles commencent par un point. Le nom de variable peut à son tour comporter plusieurs éléments. Il est possible d'entrer les adresses directement.

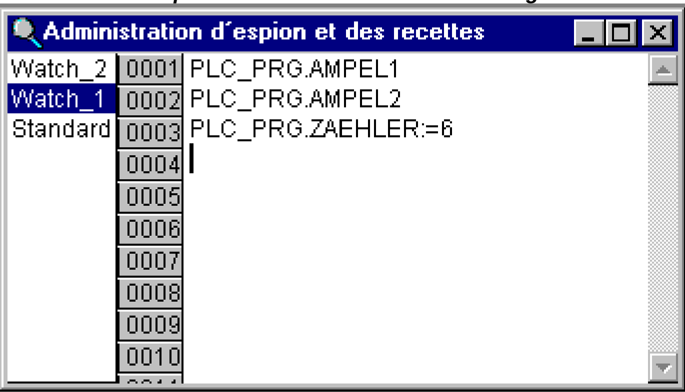
Exemple de variable à plusieurs éléments:

PLC_PRG.Instance1.Instance2.Structure.Nom_composant

Exemple de variable globale:

.global1.component1

Gestionnaire d'espion et de recettes en mode hors ligne



Des valeurs constantes peuvent être préaffectées aux variables de la liste d'espion, autrement dit, en mode En Ligne, ces valeurs peuvent être écrites dans les variables au moyen de la commande 'Extras' '**Ecrire la recette**'. Pour ce faire, la valeur constante doit être affectée à la variable par les signes ":=":

Exemple :

```
PLC_PRG.TIMER:=50
```

Dans l'exemple, la valeur 6 est préaffectée à la variable PLC_PRG.COUNTER.

6.8.2 Gestionnaire d'espion et de recettes en mode autonome

'Insérer' 'Nouvelle liste d'espion'

Cette commande permet d'insérer une nouvelle liste d'espion dans le gestionnaire d'espion et de recettes. Entrez le nom que vous souhaitez attribuer à la liste d'espion, dans la boîte de dialogue qui apparaît.

'Extras' 'Renommer liste d'espion'

Cette commande permet de modifier le nom d'une liste d'espion dans le gestionnaire d'espion et de recettes.

Entrez dans la boîte de dialogue qui apparaît le nouveau nom de la liste d'espion.

'Extras' 'Charger l'histogramme'

Cette commande permet de charger à nouveau un histogramme stocké. La boîte de dialogue pour l'ouverture des fichiers s'affiche. Sélectionnez un fichier avec l'extension "*.trc".

La commande 'Extras' 'Stocker l'histogramme' permet de stocker un histogramme.

'Extras' 'Enregistrer liste d'espion'

Cette commande permet d'enregistrer une liste d'espion. La boîte de dialogue pour enregistrer un fichier s'ouvre. Le nom du fichier est prédéfini avec le nom de la liste d'espion, auquel vient s'ajouter l'extension "*.wtc".

La liste d'espion enregistrée peut être à nouveau chargée au moyen de la commande 'Extras' 'Charger liste d'espion'.

'Extras' 'Charger liste d'espion'

Cette commande permet de charger à nouveau une liste d'espion enregistrée. La boîte de dialogue pour l'ouverture des fichiers s'affiche. Sélectionnez le fichier voulu parmi ceux qui possèdent l'extension "*.wtc". Dans la boîte de dialogue qui s'affiche, vous pouvez attribuer un nouveau nom à la liste d'espion. Le nom de fichier a été prédéfini sans extension.

La commande 'Extras' 'Enregistrer liste d'espion' permet d'enregistrer une liste d'espion.

6.8.3 Gestionnaire d'espion et de recettes en mode En Ligne

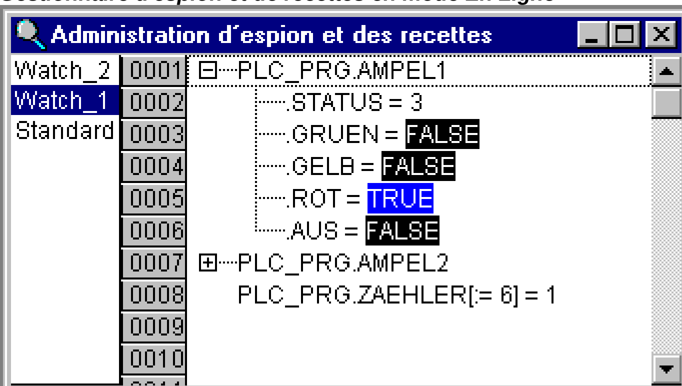
Dans le mode En Ligne, les valeurs des variables entrées sont affichées.

Les valeurs structurées (tableaux, structures ou instances de bloc fonctionnel) sont signalées par un signe plus placé devant l'identificateur. En cliquant avec la souris sur le signe plus ou en appuyant sur la touche <Entrée>, la variable est affichée ou masquée.

Si une variable de bloc fonctionnel est marquée dans la liste d'espionnage, les options 'Zoom' et 'Ouvrir l'instance' seront ajoutées au menu contextuel correspondant.

Pour entrer de nouvelles variables, il est possible d'arrêter l'affichage en utilisant la commande 'Extras' 'Espionnage actif'. Cette même commande permet d'activer l'affichage des valeurs dès que les variables ont été entrées.

Gestionnaire d'espion et de recettes en mode En Ligne



En *mode hors ligne*, des valeurs constantes peuvent être affectées à des variables (en entrant après la variable :=<Valeur>). Dans le *mode En Ligne*, ces valeurs peuvent maintenant être écrites dans les variables en utilisant la commande 'Extras' 'Ecrire la recette'.

La commande 'Extras' 'Lire la recette' permet de remplacer la valeur préaffectée à une variable par sa valeur actuelle.

Remarque : Seules sont chargées les valeurs appartenant à une liste d'espion sélectionnée dans le gestionnaire d'espion et de recettes!

'Extras' 'Espionnage actif'

Dans le mode En Ligne, cette commande permet d'arrêter ou d'activer l'affichage dans le gestionnaire d'espion et de recettes. Lorsque l'affichage est activé, un crochet (✓) apparaît devant l'élément de menu correspondant.

Pour entrer de nouvelles variables ou pour préaffecter une valeur à une variable (voir mode hors ligne), il est nécessaire d'arrêter l'affichage au moyen de cette même commande. Cette même commande permet d'activer l'affichage des valeurs dès que les variables ont été entrées.

'Extras' 'Ecrire la recette'

Lorsque le gestionnaire d'espion et de recettes travaille en mode *En Ligne*, cette commande permet d'écrire les valeurs prédéfinies (voir mode hors ligne) dans les variables.

Remarque : Seules sont chargées les valeurs appartenant à une liste d'espion sélectionnée dans le gestionnaire d'espion et de recettes!

'Extras' 'Lire la recette'

Lorsque le gestionnaire d'espion et de recettes travaille en mode *En Ligne*, cette commande permet de remplacer les valeurs prédéfinies des variables (voir mode hors ligne) par leurs valeurs actuelles.

Exemple :

```
PLC_PRG.Counter [:= <valeur actuelle>] = <valeur actuelle>
```

Remarque : Seules sont chargées les valeurs appartenant à une liste d'espion sélectionnée dans le gestionnaire d'espion et de recettes!

Forcer les valeurs des variables

Le gestionnaire d'espion et de recettes propose également les commandes 'Forcer valeurs des variables' et 'Ecrire valeur des variables'. En cliquant sur la valeur d'une variable, vous accédez à la boîte de dialogue qui permet d'entrer la nouvelle valeur pour cette variable. Les variables modifiées sont affichées en rouge dans le gestionnaire d'espion et de recettes.

6.9 Histogramme

6.9.1 Apercu et Configuration

L'enregistrement des histogrammes est disponible dans CoDeSys dès que l'option correspondante est activée dans la configuration du système cible (catégorie 'Généralités').

L'enregistrement de histogrammes assure l'enregistrement de l'évolution de la valeur des variables dans un intervalle de temps déterminé. Ces valeurs sont enregistrées dans une mémoire en anneau (tracebuffer). Si la mémoire est remplie, alors les valeurs les plus "anciennes" sont écrasées, en commençant par le début.

Un maximum de 20 variables peuvent être enregistrées simultanément. Un maximum de 500 valeurs peuvent être enregistrées pour chaque variable.

Etant donné que la dimension du tracebuffer à l'intérieur de l'automate programmable est fixe, il est possible que le nombre de valeurs enregistrées soit inférieur à 500, dans le cas de variables très nombreuses ou très grandes (DWORD).

Exemple : si 10 variables WORD doivent être enregistrées et si la mémoire à l'intérieur de l'automate programmable a une capacité de 5000 octets, alors 250 valeurs peuvent être enregistrées pour chaque variable.

Pour permettre l'enregistrement d'un histogramme, ouvrez l'objet **Histogramme** dans l'onglet **Ressources** de l'Organisateur d'objets. Créez ou chargez une configuration d'histogramme correspondante et définissez les variables d'histogramme qui doivent être enregistrées (voir 'Extras' 'Configuration de l'histogramme' et 'Sélection des variables à visualiser').

Après la création de la configuration de la histogramme et démarré l'enregistrement dans l'automate, au moyen de 'Démarrer l'histogramme', les valeurs des variables sont enregistrées. 'Lire l'histogramme' entraîne l'extraction des dernières valeurs enregistrées ainsi que la visualisation de ces valeurs sous forme de courbes dans un graphique.

Un enregistrement d'histogrammes (valeurs de variables et configuration) peut être sauvegardé et à nouveau chargé dans le format du projet (*.trc) ou en format XML (*.mon). Le paramétrage de la configuration peut être sauvegardé et à nouveau chargé dans un fichier *.tcf.

Plusieurs enregistrements peuvent être à disposition pour affichage dans le projet. Il sont repris dans une liste de sélection ('Histogramme') dans le coin supérieur droit de la fenêtre d'histogramme. La configuration de l'histogramme devant être actuellement utilisé peut être sélectionné hors de cette liste.

'Extras' 'Configuration de l'histogramme'

Cette commande vous permet d'obtenir une boîte de dialogue pour entrer les variables à enregistrer ainsi que les diverses options pour l'enregistrement de l'histogramme. La boîte de dialogue peut également être ouverte en double-cliquant sur la surface grise de la boîte de dialogue Enregistrement des histogrammes.

Boîte de dialogue Configuration de l'histogramme

Configuration de l'histogramme

Options

Variable déclencheur:

Position du déclenceur [%]: 50

Niveau du déclencheur: 0

Front du décloencheur: positive négative les deux aucun

Numéro d'échantillonages: 499

Taux d'échantillonnage [ms]: 0

Enregistrement: Single Permanent Manuel

Fermer

Annuler

Enregistrer

Charger

Variables

PLC_PRG.AMPEL1.GRUEN
PLC_PRG.AMPEL1.ROT
PLC_PRG.AMPEL1.GELB
PLC_PRG.AMPEL1.STATUS
PLC_PRG.AMPEL1.GRUEN
PLC_PRG.AMPEL1.ROT
PLC_PRG.AMPEL1.GELB
PLC_PRG.AMPEL1.STATUS

Insérer

Effacer

Manager de l'Aide

Entrée de variables de l'histogramme:

Attribuez tout d'abord un nom (Nom d'histogramme) à la configuration. Elle apparaîtra ensuite sous ce nom dans la liste de sélection 'Histogramme' dans le coin supérieur droit de la fenêtre 'Enregistrement des histogrammes', dès que le dialogue de configuration est confirmé avec OK puis refermé.

Le champ Commentaire vous permet en outre d'insérer un texte à votre gré.

Dans un premier temps, la liste des **Variables** à enregistrer est vide. Pour qu'une variable puisse y être ajoutée, celle-ci doit être entrée dans le champ situé en dessous de la liste. Ensuite, la variable peut être ajoutée à la liste au moyen du bouton **Insérer** ou de la touche <Entrée>. Pour ce faire, vous

pouvez également utiliser la Liste de sélection pour l'édition. L'utilisation de variables d'énumération est également possible.

Une variable est enlevée de la liste en la sélectionnant et en appuyant ensuite sur le bouton **Effacer**.

Une variable booléenne ou une variable analogique (ou encore une variable d'énumération) peut être entrée dans le champ **Variable déclencheur**. Pour ce faire, vous pouvez également utiliser la liste de sélection pour l'édition. La variable déclencheur sert à décrire la condition de terminaison de l'histogramme.

Dans **Niveau du déclencheur**, spécifiez la valeur de la variable déclencheur analogique pour laquelle un événement déclencheur se produit. Cette valeur peut également être indiquée par une constante ENUM.

Si l'option **positive** a été activée dans **Front du déclencheur**, alors l'événement déclencheur survient après un front montant pour une variable déclencheur de type booléen ou encore si une variable déclencheur analogique franchit le niveau du déclencheur, du bas vers le haut. De façon analogue, si l'option **négative** a été activée, le déclenchement intervient après un front descendant ou lors d'un franchissement du haut vers le bas. Si l'option **les deux** a été activée, alors le déclenchement intervient après un front descendant et un front montant, ou encore après un franchissement positif et un franchissement négatif. Enfin, si l'option **aucun** a été activée, aucun événement déclencheur ne survient.

Dans **Position du déclencheur**, spécifiez quel pourcentage des valeurs de mesure doit être enregistré avant l'événement déclencheur. Par exemple, si vous entrez 25, cela signifie que 25 % des valeurs de mesure sont visualisées avant l'événement déclencheur et 75 % après l'événement déclencheur, après quoi l'histogramme est terminé.

Dans le champ **Taux d'échantillonnage**, vous pouvez spécifier l'intervalle de temps qui sépare deux enregistrements en millisecondes, ou en microsecondes si le système cible le supporte. La préaffectation "0" signifie qu'un processus d'échantillonnage est effectué par cycle.

Choisissez le mode d'appel des valeurs enregistrées. Si **Single** est activé, alors le nombre de mesures spécifié dans le champ **Nombre** est visualisé une fois. Si **Permanent** est activé, l'extraction de l'enregistrement des valeurs de mesure, pour le nombre de mesures spécifié, se répète continuellement. Par exemple, si vous entrez "35" dans Nombre, cela signifie que la première visualisation comporte les premières valeurs de mesure, de la 1ère à la 35ème. Ensuite, l'enregistrement des 35 valeurs de mesure suivantes est appelé (36ème -70ème), et ainsi de suite. Si l'option **Manuel** est activée, l'extraction de l'enregistrement de l'histogramme est réalisée à la demande, au moyen de la commande 'Extras' 'Lire l'histogramme'.

Le mode d'appel fonctionne indépendamment du fait qu'une variable déclencheur a été définie ou non. Si aucune variable déclencheur n'est spécifiée, alors le tracebuffer est rempli avec le nombre spécifié de valeurs de mesures et le contenu du tampon est lu et visualisé lors de l'appel.

Le bouton **Enregistrer** permet d'enregistrer dans un fichier la configuration de l'histogramme créée. Pour ce faire, accédez à la boîte de dialogue standard 'Fichier enregistrer sous'.

Le bouton **Charger** vous permet de charger à nouveau une configuration de l'histogramme placée en mémoire. Pour cela, vous obtenez la boîte de dialogue standard 'Ouvrir fichier'.

Remarque : Nous attirons votre attention sur le fait que les boutons **Enregistrer** et **Charger** de la boîte de dialogue Configuration de l'histogramme s'appliquent uniquement à la configuration et non aux valeurs d'un enregistrement de l'histogramme (contrairement aux options de menu 'Extras' 'Stocker l'histogramme' et 'Extras' 'Charger l'histogramme').

Si le champ '**Variable déclencheur**' est vide, alors l'enregistrement de l'histogramme se poursuit indéfiniment. Il peut être terminé expressément à l'aide de la commande 'Extras' 'Arrêter l'histogramme'.

Remarque : Si une configuration de tâche est utilisée pour la commande de l'exécution du programme, la fonction d'histogramme se rapporte à la Tâche de débogage (voir chapitre 6.6, Configuration des tâches).

Sélection des variables à visualiser

Les zones de liste modifiable situées à droite de la fenêtre de visualisation des courbes contiennent chacune toutes les variables d'histogramme définies dans la configuration de l'histogramme. Après qu'une variable ait été sélectionnée dans la liste et que le tracebuffer ait été lu, cette variable est représentée dans la couleur correspondante (Var 0 vert etc.). Les variables peuvent aussi être sélectionnées lorsque des courbes ont déjà été affichées.

Un maximum de huit variables peuvent être observées simultanément dans la fenêtre de l'histogramme.

'Extras' 'Lancer l'histogramme'

Icône : 

Cette commande permet de transférer la configuration de l'histogramme vers l'automate et de démarrer l'enregistrement de l'histogramme dans l'automate programmable.

'Extras' 'Lire l'histogramme'

Icône : 

Cette commande permet d'extraire le tracebuffer actuel de l'automate programmable et de visualiser les valeurs des variables sélectionnées.

'Extras' 'Lecture automatique de l'histogramme'

Cette commande permet d'extraire automatiquement le tracebuffer actuel de l'automate programmable et de visualiser continuellement les valeurs des variables sélectionnées.

Si la lecture du tracebuffer se fait automatiquement, alors un crochet est mis devant l'élément de menu correspondant.

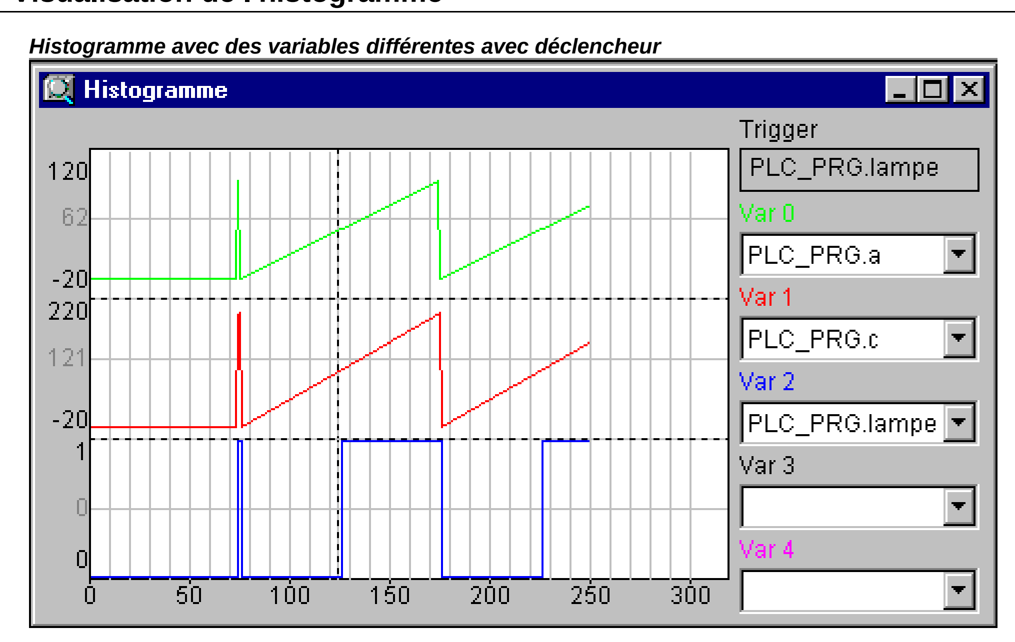
'Extras' Arrêter l'histogramme'

Icône :

Cette commande arrête l'enregistrement de l'histogramme dans l'automate programmable.

6.9.2 Visualisation de l'histogramme

Histogramme avec des variables différentes avec déclencheur



Le coin supérieur droit ('Histogramme') de la fenêtre d'histogramme reprend le nom de la configuration d'histogramme actuellement utilisée, et un commentaire éventuellement disponible est affiché dans le coin inférieur droit.

Dans le cas où un tracebuffer est chargé, les valeurs de toutes les variables à visualiser sont extraites du tracebuffer et sont visualisées. Si le champ Taux d'échantillonnage n'a pas fait l'objet d'une configuration, alors l'axe des X est gradué avec les numéros d'échantillonnage successifs des valeurs enregistrées. L'affichage de l'état de l'enregistrement d'histogramme indique si le tracebuffer n'est pas encore rempli ou si l'enregistrement de l'histogramme est terminé.

Si une valeur a été spécifiée pour le taux d'échantillonnage, alors l'axe des X indique le temps d'acquisition des valeurs de mesure enregistrées. La valeur de mesure enregistrée la plus "ancienne" reçoit le temps 0. Dans l'exemple, les valeurs des 25 dernières secondes sont affichées.

L'axe des Y est gradué avec des valeurs ayant le type de données approprié. La graduation est disposée de façon à ce que la valeur la moins élevée et la valeur la plus élevée rentrent dans les limites de la fenêtre. Dans l'exemple, Var 0 possède comme valeur minimale 0 et comme valeur maximale 100, d'où la configuration de la graduation au niveau du bord gauche.

Si la condition du déclencheur est vérifiée, alors une ligne verticale discontinue est dessinée entre les valeurs avant vérification de la condition et les valeurs après vérification de la condition.

Une mémoire qui a été lue, est conservée aussi longtemps que l'on ne change pas de projet et que l'on ne quitte pas le système.

'Extras' 'Afficher curseur'

La méthode la plus rapide pour placer un curseur dans la fenêtre graphique est de cliquer avec la touche gauche de la souris dans la fenêtre. Le curseur peut être déplacé à volonté à l'aide de la souris. Au-dessus de la fenêtre du graphique, vous pouvez lire la position actuelle du curseur sur l'axe des X. A côté des textes Var0, Var1, …, VarN, la valeur de la variable correspondante est affichée.

Vous pouvez parvenir au même résultat en sélectionnant la commande **'Extras' 'Afficher curseur'**. Avec cette commande, deux droites verticales apparaissent dans l'histogramme qui se trouvent d'abord l'une sur l'autre. Vous pouvez déplacer une de ces lignes vers la droite ou vers la gauche, au moyen des touches directionnelles. En appuyant sur <Ctrl>+<gauche> ou sur <Ctrl>+<droite>, selon le cas, vous pouvez accélérer le déplacement d'un facteur 10.

En maintenant en plus la touche <Maj> enfoncée, vous pouvez déplacer l'autre droite et la différence par rapport à la première droite est indiquée.

'Extras' 'Multivoie'

Cette commande permet de basculer entre la visualisation monovoie et la visualisation multivoie de l'histogramme. Dans le cas d'une visualisation multivoie, un crochet est mis devant l'élément de menu.

La visualisation multivoie est choisie comme configuration par défaut. Dans ce cas, la fenêtre de visualisation est scindée en huit parties pour les courbes à visualiser. Au niveau du bord, la valeur maximale et minimale pour chaque courbe est affichée sous forme de graduation.

Dans le cas d'une visualisation monovoie, toutes les courbes ont la même graduation et sont visualisées de manière superposée. Cela peut être utile pour visualiser des écarts entre les courbes.

'Extras' 'Montrer la matrice'

Cette commande permet d'activer ou de désactiver l'affichage de la grille dans la fenêtre d'espion. Lorsque l'option est activée, un crochet apparaît devant l'option.

'Extras' 'Graduation Y'

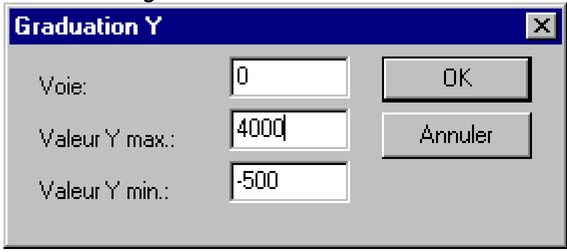
Cette commande permet de modifier la graduation prédéfinie de l'axe des Y d'une courbe dans la visualisation de l'histogramme. En double-cliquant sur une courbe, vous avez accès au dialogue 'Graduation Y'.

Tant que l'option Automatique est activée, on utilise la graduation par défaut, selon le type de variable concernée. Dans le cas d'énumérations, les valeurs énumératives correspondantes sont affichées

comme libellé d'échelle. Pour changer la graduation, désactivez l'option 'Automatique', puis entrez dans la boîte de dialogue le numéro de la courbe de votre choix (**Voie**) et la nouvelle valeur maximale (**Valeur Y max.**) et la nouvelle valeur minimale (**Valeur Y min.)** sur l'axe des Y.

Vous pouvez également accéder à ce dialogue en double-cliquant sur une courbe. La voie et les valeurs actuelles de graduation sont affichées.

Boîte de dialogue Graduation Y



'Extras' 'Etendre'

Icône :

Cette commande permet d'étirer (d'effectuer un zoom) les valeurs affichées. La position initiale peut être ajustée au moyen de la barre de défilement horizontale. Lorsque cette commande est exécutée plusieurs fois de suite, l'extrait d'histogramme affiché est de plus en plus petit.

Cette commande réalise le contraire de la commande 'Extras' 'Compresser'.

'Extras' 'Compresser'

Icône :

Cette commande permet de compresser les valeurs affichées dans l'histogramme. Autrement dit, cette commande permet de suivre l'évolution des variables d'histogramme pendant une durée plus grande. Il est possible de répéter cette commande.

Cette commande réalise le contraire de la commande 'Extras' 'Etendre'.

6.9.3 Enregistrer les valeurs de l'histogramme

Les commandes de ce menu servent à enregistrer la configuration et les valeurs d'un enregistrement d'histogramme dans un fichier au format du projet, ou encore de charger ces valeurs à partir d'un tel fichier. L'enregistrement peut en outre être sauvegardé dans un fichier ASCII.

Remarque: Notez les possibilités alternatives de sauvegarde et de chargement du menu 'Extras' 'Configurations de l'histogramme externes' (format XML, fichier *.mon)!

'Extras' 'Enregistrer les valeurs de l'histogramme' 'Enregistrer les valeurs de l'histogramme'

Cette commande permet de stocker un histogramme. La boîte de dialogue pour enregistrer un fichier s'ouvre. L'extension "*.trc" est ajoutée au nom du fichier.

Notez que dans ce cas, les valeurs de mesure et la configuration de l'histogramme sont enregistrées dans le format du projet, alors que l'enregistrement au sein du dialogue de configuration ne concerne que la configuration elle-même.

Notez en outre que les valeurs de mesure et la configuration peuvent être enregistrées dans un fichier au format XML, voir à ce sujet le menu 'Extras' 'Configurations de l'histogramme externes'.

L'histogramme enregistré peut être à nouveau chargé au moyen de la commande 'Extras' "Enregistrer les valeurs de l'histogramme" ' 'Extras' 'Charger l'histogramme'.

Remarque : Notez les possibilités alternatives de sauvegarde par le biais des commandes du menu 'Extras' 'Configurations de l'histogramme externes'.

'Extras' 'Enregistrer les valeurs de l'histogramme' 'Charger l'histogramme'

Cette commande permet de charger à nouveau un histogramme stocké. La boîte de dialogue pour l'ouverture des fichiers s'affiche. Sélectionnez un fichier avec l'extension "*.trc".

L'enregistrement est représenté dans la fenêtre d'histogramme et la configuration est reprise comme configuration actuelle dans le projet.

La commande 'Extras' 'Enregistrer les valeurs de l'histogramme' 'Enregistrer les valeurs de l'histogramme' permet de stocker un histogramme.

'Extras' 'Valeurs dans le fichier ASCII'

Cette commande permet de stocker un histogramme dans un fichier ASCII. La boîte de dialogue pour enregistrer un fichier s'ouvre. L'extension "*.txt" est ajoutée au nom du fichier. Les valeurs sont enregistrées dans le fichier, selon le schéma suivant:

```
CoDeSys Trace
D:\CODESYS\PROJECTS\FEUX.PRO
Cycle PLC_PRG.COUNTER PLC_PRG.LIGHT1
0 2 1
1 2 1
2 2 1
.....
```

Si, dans la configuration de l'histogramme, le champ Taux d'échantillonnage n'a pas été défini, alors le cycle figure dans la première colonne, c.-à-d. qu'une seule acquisition a eu lieu par cycle. Dans le cas contraire, la première colonne affiche le moment (en ms) auquel ont été enregistrées les valeurs depuis le démarrage de l'enregistrement de l'histogramme.

Dans les colonnes suivante sont enregistrées les valeurs correspondantes des variables d'histogramme. Les valeurs sont séparées entre elles par un espace.

Les noms de variable correspondants sont visualisés au niveau de la troisième ligne, l'un à côté de l'autre et selon leur ordre de succession (PLC_PRG.COUNTER, PLC_PRG.LIGHT1).

6.9.4 Configurations de l'histogramme externes

'Extras' 'Configurations de l'histogramme externes

Les commandes de ce menu servent à enregistrer les configurations et les valeurs de l'histogramme, ou encore de charger ces mêmes valeurs dans le projet à partir de fichiers ou de l'automate programmable. En outre, une de ces configurations peut être définie comme étant celle à utiliser dans le projet.

Remarque: Notez les possibilités alternatives de sauvegarde et de chargement du menu 'Extras' 'Enregistrer les valeurs de l'histogramme' (format du projet, fichier *.trc, ASCII)!

Enregistrer dans le fichier

Cette commande permet d'enregistrer un histogramme (configuration + valeurs) dans un fichier au format XML. Pour ce faire, vous avez accès au dialogue d'enregistrement de fichier. L'extension de fichier *.mon est automatiquement utilisée.

Un fichier *.mon peut être chargé dans le projet via la commande 'Charger du fichier'.

Charger du fichier

Cette commande permet de charger dans le projet un histogramme (configuration + valeurs) présent dans un fichier au format XML. Pour ce faire, le dialogue d'ouverture des fichiers recherche automatiquement les fichiers avec l'extension *.mon. L'enregistrement d'histogramme chargé est affiché dans la fenêtre d'histogramme et est ajouté à la liste de sélection dans le champ 'Histogramme' du dialogue de configuration. Pour en faire la configuration actuelle de projet, il faut sélectionner la commande 'Reprendre en tant que configuration du projet'.

Il est possible de créer un fichier *.mon via la commande 'Enregistrer dans le fichier'.

Remarque : Notez les possibilités alternatives de sauvegarde et de chargement du menu 'Extras' 'Enregistrer les valeurs de l'histogramme' (format du projet, fichier *.trc, ASCII)!

Enregistrer dans l'automate

Cette commande permet, en mode En ligne, d'enregistrer sur l'automate programmable un enregistrement d'histogramme qui se trouve dans un fichier de format XML. Pour ce faire, vous avez accès au dialogue standard de sélection de fichier, affichant tout d'abord et par défaut les fichiers avec extension *.mon. Notez à cet égard la possibilité d'enregistrer des configurations d'histogrammes en format XML dans de tels fichiers *.mon ('Extras' 'Enregistrer dans le fichier').

Voir en outre : 'Charger de l'automate'.

Charger de l'automate

Cette commande permet de charger dans le projet l'histogramme (configuration + valeurs, fichier au format XML) se trouvant actuellement sur l'automate programmable. Il est affiché dans la fenêtre d'histogramme et peut être repris dans la configuration actuelle du projet.

Voyez également à cet effet : 'Enregistrer dans l'automate'

Appliquer configuration du projet

Cette commande vous permet de reprendre dans le projet la configuration d'histogramme (sélectionnée à l'instant dans la fenêtre de sélection 'Histogramme' du dialogue de configuration) comme étant la configuration actuellement active. La liste de sélection vous propose, outre la configuration momentanément active (à la position supérieure), toutes les autres configurations qui ont déjà été chargées dans le projet avec la commande 'Charger du fichier' à partir de fichiers *.mon (pour affichage par exemple).

6.10 Environnement de travail

Ce noeud dans le registre 'Ressources' contient une représentation des options de projet configurées. (voir chapitre 4.2, options de projet). S'il est ouvert, le dialogue **Options** apparaît à l'écran, avec ses catégories connues.

6.11 Manager des paramètres

Apercu

Le Gestionnaire des paramètres est un composant du système de programmation **CoDeSys** qui est spécifique au système cible, et il doit être activé dans la configuration du système cible. (voir chapitre 6.12).

Le Gestionnaire des paramètres peut être utilisé pour avoir accès aux **paramètres** à partir de tous les systèmes compatibles CoDeSys dans le réseau, cela à des fins d'**échange de données** (normalement par bus de champ). Pour ce faire, il est possible de créer et de traiter des listes de paramètres dans l'éditeur ; on peut également les charger à partir du ou sur le système cible.

Remarque: Les listes de paramètres peuvent également être créées ou complétées directement par des instructions de Pragma au sein de déclarations.

<u>Que sont les paramètres ? :</u>

À ce niveau, il faut distinguer différents types de paramètres :

- Variables de processus du projet CoDeSys CEI
- Paramètres indépendants de processus

- Paramètres système spécifiques, prédéfinis via le système cible
- Instances de blocs fonctionnels ou variables structurées, variables de type array

Chaque paramètre est défini par un jeu défini d'**attributs**, comme p.ex. 'Valeur', 'Valeur par défaut', 'Droits d'accès', et tout spécialement par une **clé d'accès univoque** ('Index', 'Sous-index', 'Nom'), laquelle donne accès à l'entrée dans la liste des paramètres afin de pouvoir lire ou écrire des données. Cet échange de données peut se faire via les **services de communication**, et il n'est pas nécessaire de connaître l'adresse des variables ou d'utiliser des fonctions supplémentaires. Ainsi, l'utilisation du gestionnaire des paramètres représente une alternative fonctionnelle à l'utilisation de variables réseau.

<u>Que sont les listes de paramètres ? :</u>

Les **listes de paramètres** servent à la gestion des paramètres, peuvent être enregistrées dans le projet et être chargées dans le système cible actuellement rattaché au programme CEI. À chaque type de paramètres (voir ci-dessus) correspond un Type de liste.

Chaque entrée de paramètres est représentée sur une ligne dans une liste de paramètres. Chaque **colonne** de la liste représente un des attributs du paramètre (p.ex. Index, Valeur par défaut, ...). En plus d'un jeu défini d'attributs standard, on dispose également, pour la description d'un paramètre, d'attributs spécifiques aux fabricants.

Les différentes définitions au sein d'un **fichier descriptif spécifique au système cible** (odconfig.xml) déterminent les attributs (= colonnes dans l'éditeur de gestionnaire des paramètres) visibles et éditables, ainsi que leur classification dans la liste des paramètres. S'il n'y a pas de fichier descriptif, le jeu standard et complet d'attributs sera affiché, prédéfinis avec les valeurs standard.

Outre des listes de variables de projet et de constantes de projet, le Gestionnaire des paramètres peut également gérer des listes de paramètres système. Ces dernières sont définies de manière fixe par le système cible. En outre, on peut créer des listes d'arrays, d'instances de blocs fonctionnels ou de variables structurées, ces listes reposant sur des **Modèles** définis par l'utilisateur, ces derniers pouvant également être créés dans le Gestionnaire des paramètres.

Comme les données sont gérées indépendamment du programme CEI, on peut utiliser une liste de paramètres afin par exemple d'enregistrer des 'Recettes' qui seraient gardées si le programme est remplacé par une version de programme différente. En outre, un automate programmable en cours pourrait être alimenté de différentes 'Recettes' sans que cela ne nécessite un téléchargement de programme.

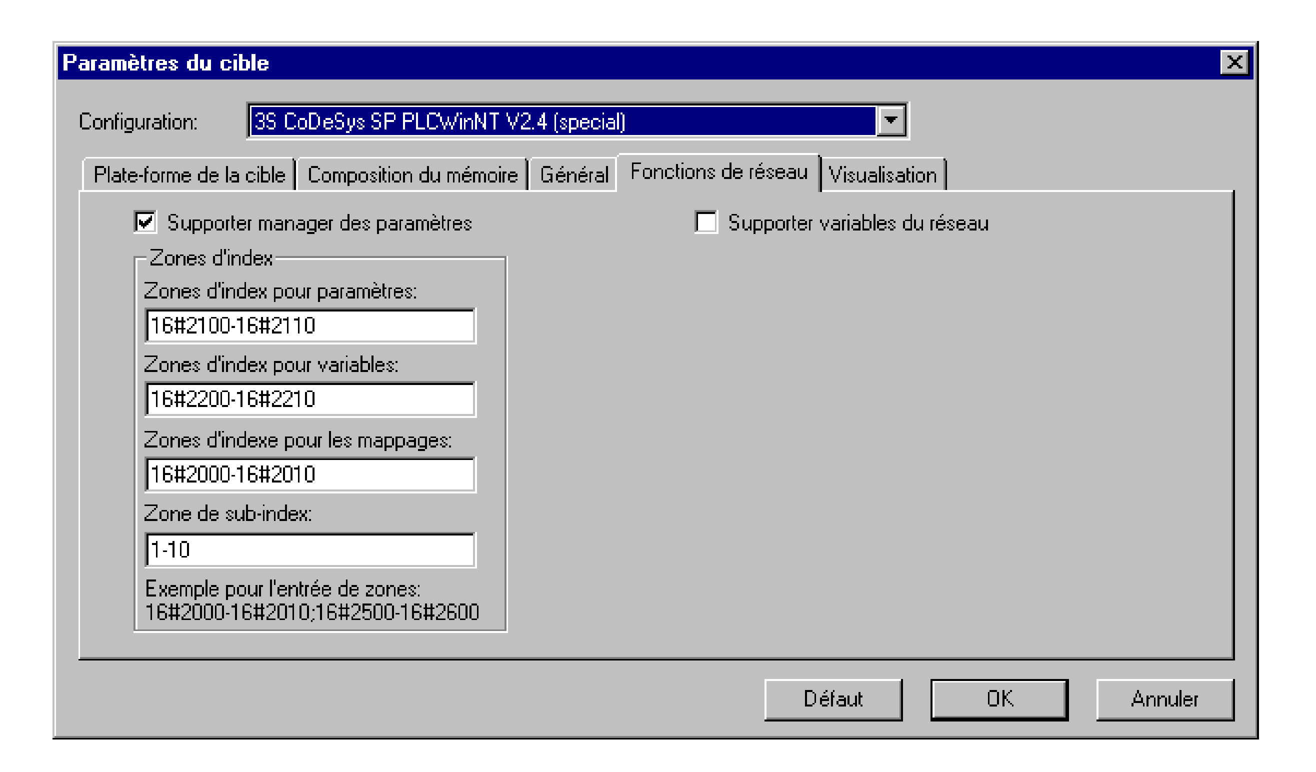
Remarque : Le système cible détermine si les contenus du Gestionnaire des paramètres doivent être repris dans un **projet d'initialisation** lors de la création de ce dernier.

6.11.1 Manager des paramètres, Activer

Le Gestionnaire des paramètres doit être activé dans la **configuration du système cible,** catégorie **Fonctions réseau** (voir chapitre 6.12)**.**

Il faut également définir ici les zones d'index et de sous-index pour les entrées dans les listes de paramètres, cela pour les listes de paramètres du type paramètres, variables, et - si supporté par le système cible - mappages (pour CAN Device PDO).

Le système cible détermine dans quelle mesure l'utilisateur peut consulter ou éditer ces réglages.



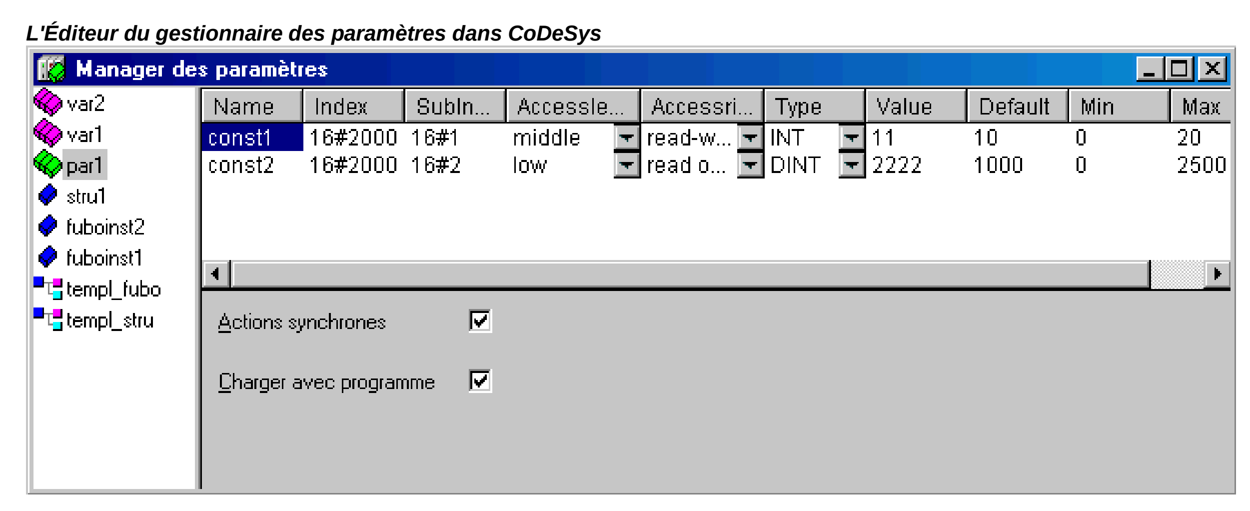
6.11.2 Manager des paramètres, Editeur

Aperçu

Pour ouvrir l'Éditeur, sélectionnez l'entrée 'Gestionnaire des paramètres' à l'onglet Ressources dans CoDeSys. Vous pouvez ici créer, éditer et enregistrer des listes de paramètres, et échanger celles-ci en mode En Ligne avec l'automate programmable actuellement connecté.

Remarque: Afin de disposer de la fonctionnalité du gestionnaire des paramètres dans le projet CoDeSys, l'option 'Supporter gestionnaire des paramètres' doit être activée dans la configuration du système cible, et les zones d'index et de sous-index correspondantes doivent être définies !

L'Éditeur du gestionnaire des paramètres dans CoDeSys



La fenêtre d'éditeur est divisée en deux. La partie gauche sert à la navigation, et elle affiche les listes de paramètres actuellement chargées dans le gestionnaire des paramètres. La partie droite contient un éditeur de listes, et les colonnes sont libellées avec les noms des attributs de paramètre.

Dans la **fenêtre de navigation**, vous pouvez insérer, effacer, trier ou renommer des listes de paramètres de différents types (variables, paramètres (constantes), modèles, instances, paramètres système). .

Dans **l'éditeur de listes**, vous devez insérer une ligne par entrée de paramètres, cette ligne reprenant les attributs du paramètre. Chaque type de liste dispose d'une gamme spéciale d'attributs (colonnes) qui sont soit éditables, soit simplement affichés. Cette gamme d'attributs peut être définie par un **fichier descriptif spécifique au système cible** ; on utilise sinon le réglage standard.

La touche <F6> vous permet de passer de la fenêtre de navigation à l'éditeur de listes et vice-versa.

Remarque: Les listes de paramètres peuvent également être créées ou complétées directement par des instructions de Pragma au sein de déclarations.

En **mode En ligne**, vous pouvez charger les listes créées dans l'éditeur sur le système cible actuellement connecté, ou bien vous pouvez avoir accès aux listes de paramètres présentes dans le système cible afin de procéder à des échanges de données avec d'autres systèmes au sein du réseau (téléchargement en amont, écriture de valeurs). En outre, la fenêtre d'éditeur affiche les valeurs actuelles d'exécution des paramètres (espionnage).

Tant qu'il n'y a pas communication établie avec un système cible, il est seulement possible de créer des listes de paramètres et de les enregistrer localement au niveau du projet.

6.11.3 Listes de paramètres: Types et Attributs

Le gestionnaires des paramètres peut gérer les types de listes de paramètres ci-dessous :

Variables: Les entrées dans des listes de paramètres de ce type représentent des variables de processus du projet.

Paramètres: Les entrées dans des listes de paramètres de ce type représentent des données dont les valeurs sont indépendantes du processus.

Paramètres du système: Les entrées dans des listes de paramètres de ce type représentent des constantes spéciales indépendantes du processus et prédéfinies par le système cible. Les listes de paramètres système ne peuvent être effacées ou renommées.

Modèle: Un modèle ne contient pas d'entrées de paramètres auxquelles on peut avoir accès directement à des fins d'échange de données. Les entrées servent plutôt à la configuration de base des composants d'un bloc fonctionnel précis ou d'une structure. Cette configuration de base peut alors être utilisée pour la création de listes de paramètres du type 'Instance'.

Instance: Les entrées dans des listes de paramètres de ce type représentent des entrées de paramètres pour des variables de type bloc fonctionnel ou structure, en d'autres termes pour des instances et des variables structurées. Afin de faciliter la création de listes d'instances, on utilise un modèle (voir plus haut) créé auparavant et également au sein du gestionnaire des paramètres.

Mappages: Ce type de liste n'est pas disponible s'il n'est pas supporté par le Système cible. Les entrées se composent de références à des variables de processus qui peuvent être "mappées"dans un CAN-Device. En principe, il s'agit d'une liste de variables, fonctionnant sur sa propre zone d'index/de sous-index. Cette zone doit être définie dans la configuration du système cible, Catégorie Fonctions de réseau ! Le CAN-Device n'utilise dans ce cas **que** les entrées dans les listes de type 'Mappage', alors que dans les autres cas, toutes les entrées des listes de variables et d'instances sont proposées dans le dialogue 'Mappage PDO par défaut' dans la configuration de l'automate.

La représentation des différents types de listes au sein de l'Éditeur du gestionnaire des paramètres dépend du système cible.

Instances et Modèles

Une liste de paramètres de type 'Instance' ...

... gère des entrées de paramètres qui représentent un **bloc fonctionnel** précis, ou encore une **variable structurée** ou un **array**. Chaque liste d'instances pour un bloc fonctionnel ou une variable structurée se base sur un modèle qui a également du être spécialement défini dans le gestionnaire des paramètres pour le bloc fonctionnel ou la structure.

Une liste de paramètres de type 'Modèle' ...

... ne contient pas d'entrées de paramètres auxquelles on peut avoir accès directement à des fins d'échange de données. On prédéfinit ici plutôt des décalages d'index ou de sous-index ainsi que certains attributs pour les entrées de paramètres qui représentent les composants d'un bloc fonctionnel précis ou d'une structure. Ce modèle peut alors être utilisé dans une liste de paramètres de type 'Instance' (voir plus haut), ce qui facilite la création de listes de paramètres pour plusieurs variables de projet représentant les instances du même bloc fonctionnel ou de la même structure.

Création d'un Modèle:

Saisissez dans le champ **POU de base** le nom du bloc fonctionnel ou de la structure pour le(la)quel(le) le modèle vaut. Vous pouvez utiliser la liste de sélection pour l'édition (<F2>) afin d'opérer une sélection parmi les modules de projet disponibles. Appuyez sur **Appliquer** afin de reprendre les composants du module sélectionné dans l'éditeur de listes. Éditez ensuite les entrées d'attributs puis refermez la liste afin qu'elle puisse être disponible pour utilisation dans une liste d'instances.

La commande **Insérer les entrées manquantes** dans le menu contextuel ou dans le menu 'Extras' provoque une actualisation des entrées à l'état actuel du module ayant servi de base (POU de base). Ceci s'avère éventuellement nécessaire ou souhaitable lorsque quelques entrées ont été effacées dans la liste ou lorsque l'on a procédé à des changements dans le module de base.

Afin de pouvoir créer des listes de paramètres d'instance pour des variables de type array, il n'est pas nécessaire de créer un modèle dans le gestionnaire des paramètres. Le type de modèle ARRAY est disponible de manière implicite.

Création d'une liste de paramètres d'instance :

Choisissez le modèle souhaité dans le champ **Modèle**. La liste de sélection vous propose tous les modèles actuellement disponibles dans le gestionnaire des paramètres pour des blocs fonctionnels et des structures, ainsi que le type de modèle 'ARRAY'.

Saisissez dans le champ **Variable de base** la variable de projet contenant les composants pour lesquels on doit créer des entrées de paramètres. Cette variable doit être du type du bloc fonctionnel, de la structure ou de l'array pour lequel le modèle sélectionné vaut.

Saisissez un **Index de base** et un **sous-index de base** pour l'instance. Les valeurs saisies ici sont à considérer comme des décalages qui sont automatiquement ajoutés aux valeurs d'index ou de sous-index définis pour chaque composant du modèle (pour les array, on part chaque fois de 0). Le résultat de l'addition est également saisi automatiquement dans le champ d'attribut 'Index' ou 'Sous-index'. Par exemple, si vous saisissez ici l'index de base "3" pour un composant, et qu'un décalage d'index de 3000 pour ce composant est défini dans le modèle, le composant sera fixé à l'index 3003.

Appuyez sur le bouton **Appliquer** afin de reprendre les composants prédéfinis dans l'éditeur de listes.

La commande **Insérer les entrées manquantes** dans le menu contextuel ou dans le menu 'Extras' provoque une actualisation des entrées à l'état actuel du modèle utilisé. Cela peut s'avérer utile lorsque des entrées ont été effacées dans la liste des paramètres ou lorsque le modèle a été changé.

Exemple de création d'une liste de paramètres d'instance:

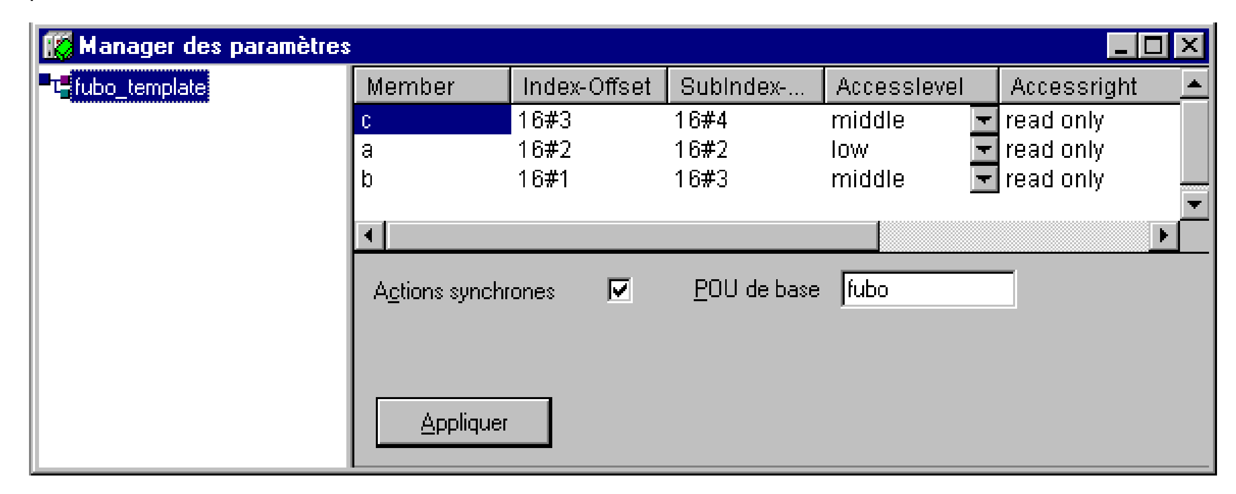
Créez un bloc fonctionnel *fubo* dans le projet avec les variables ci-dessous:*a*,*b*,*c*.

Définissez les instances de bloc fonctionnel suivantes dans *PLC_PRG*: *inst1_fubo:fubo*; *inst2_fubo:fubo*;.

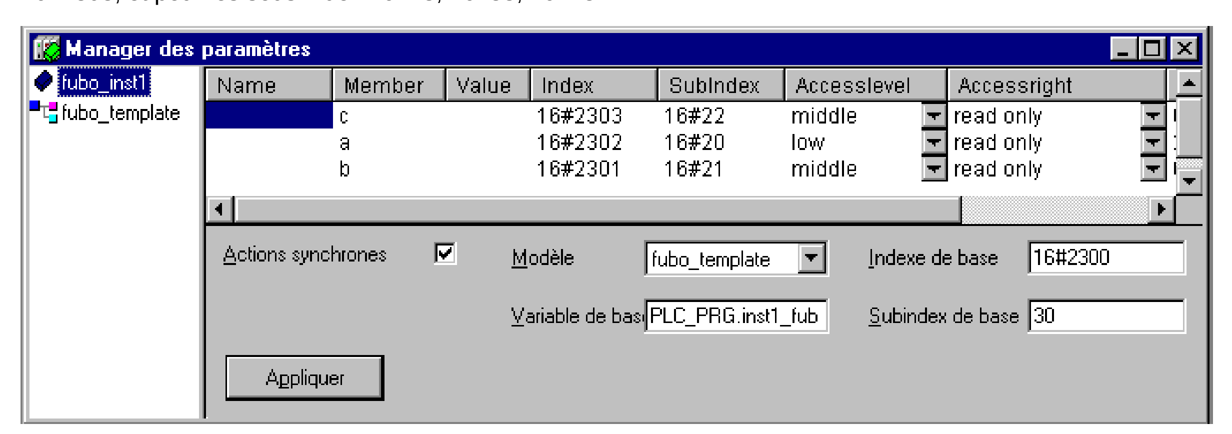
Compilez le projet.

Ouvrez le gestionnaire des paramètres afin de créer des listes de paramètres pour les variables *inst1_fubo.a, inst1_fubo.b, inst1_fubo.c* et *inst2_fubo.a, inst2_fubo.b, inst2_fubo*. Insérez d'abord pour ce faire une liste de type 'Modèle' et nommez-la "*fubo_template*". Définissez le POU de base: "*fubo*". Appuyez sur 'Appliquer' puis définissez quelques attributs pour les composants a,b,c. Saisissez par exemple

des décalages d'index: pour a: 16#1, pour b: 16#2, pour c: 16#3. De même pour les décalages de sous-index, p.ex. a: 16#2, b: 16#3, c: 16#4.



Insérez alors une nouvelle liste de paramètres de type 'Instance'. Sélectionnez le modèle "fubo_template". Saisissez la variable de base "inst1_fubo". Définissez l'index de base: p.ex. 16#2300 ainsi qu'un sous-index de base de 30 (notez les zones d'index définies dans la configuration du système cible !).Appuyez maintenant sur 'Appliquer' afin d'obtenir dans les entrées de listes les index actualisés pour les composants *a, b,* et *c*, calculés par l'addition du décalage de base et du décalage défini dans le modèle: pour les index: 16#2301, 16#2302, 16#2303; et pour les sous-index:16#23, 16#33, 16#43.



Sur base de ces entrées créées automatiquement, vous pouvez maintenant continuer à éditer la liste des paramètres.

6.11.4 Gestion des listes de paramètres

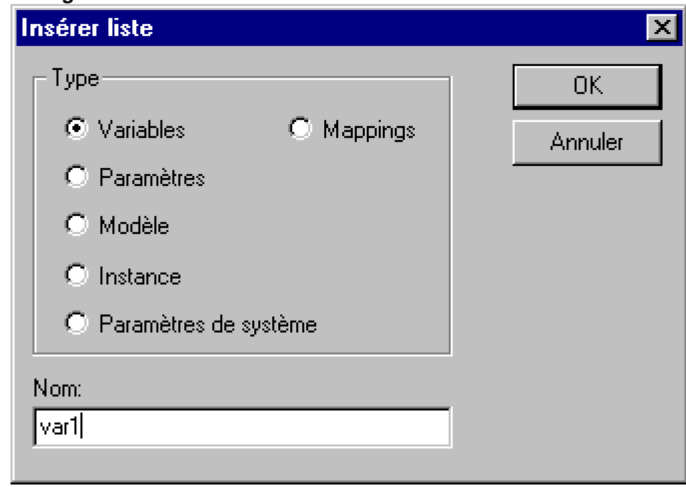
Insérer liste

Raccourci clavier : Ins

Afin d'insérer une nouvelle liste de paramètres dans le gestionnaire des paramètres, utilisez la commande 'Liste' dans le menu 'Insérer' ou encore la commande 'Insérer nouvelle liste' dans le menu contextuel. Ces commandes sont disponibles lorsque la fenêtre de navigation vide est à l'avant-plan ou lorsqu'une entrée y est déjà marquée.

Vous avez accès au dialogue 'Insérer liste':

Dialog 'Insérer liste'



Saisissez un Nom pour la nouvelle liste de paramètres (doit être univoque au sein des types de listes) puis choisissez un des types ci-dessous :

Variables	Entrées de variables de processus
Paramètres	Entrées de données dont les valeurs ne dépendent pas du processus
Modèle	Modèle pour un jeu d'attributs pour les composants d'un bloc fonctionnel ou d'une structure (pouvant être utilisé dans des listes de type 'Instance' (voir plus bas))
Instance	Entrées de variables du type bloc fonctionnel ou structure (instance), se basant sur un modèle approprié (voir plus haut)
Mappages	Entrées de variables de processus pouvant être utilisées pour le mappage PDO dans un CAN-Device. Ce type n'est disponible que si le système cible le supporte !
Paramètres du système	Entrées de paramètres dont les valeurs ne dépendent pas du processus et prédéfinies par le système cible.

Dès que les entrées ont été confirmées par **OK** et que le dialogue est refermé, la liste créée à l'instant apparaît comme entrée dans la fenêtre de navigation. Le type de liste est représenté par le symbole placé devant.

Dans l'éditeur de listes, les attributs adéquats sont affichés comme titres de colonnes. La sélection et la disposition des colonnes sont prédéfinis dans un fichier descriptif spécifique au système cible ; si un tel fichier n'existe pas, on utilise le réglage standard.

La nouvelle liste peut maintenant être éditée, cela en insérant une ligne pour chaque entrée de paramètre souhaitée.(voir chapitre 6.11.5, éditer des listes de paramètres).

Liste de paramètres\Renommer

Utilisez la commande 'Renommer liste' dans le menu 'Insérer' ou dans le menu contextuel afin renommer la liste de paramètres marquée dans le fenêtre de navigation. La commande ouvre un cadre éditer, qu'aussi est crée par un clic double sur le nom de la liste.

Couper / Copier / Insérer répertoire de listes de paramètres

Raccourci clavier : <Ctrl> + <X>, <Ctrl> + <C>, <Ctrl> + <V>

La disposition des listes de paramètres dans la fenêtre de navigation peut être modifiée par le biais des commandes 'Couper', 'Copier' et 'Insérer' (menu 'Éditer' ou menu contextuel).

La commande 'Couper ' ou 'Couper liste' stocke la liste marquée à l'instant dans une mémoire tampon; à partir de là, elle pourra à nouveau être insérée en un autre endroit via les commandes

'Insérer' ou 'Insérer liste'. Avant cette nouvelle insertion, marquez la liste se trouvant en dessous de l'endroit souhaité.

La commande 'Copier' ou 'Copier liste' utilise également cette mémoire tampon provisoire, à cette différence près que l'entrée originale est gardée et qu'une copie puisse en outre être faite dans l'arborescence de navigation par le biais de la commande 'Insérer' ou 'Insérer liste'.

Effacer Liste

Raccourci clavier : <Supp>

La liste actuellement marquée dans la fenêtre de navigation peut être supprimée avec la commande 'Effacer' (menu 'Éditer') ou 'Effacer liste' (menu 'Extras' ou menu contextuel).

Veuillez noter : En mode En ligne, cette commande efface la liste correspondante dans le système d'exécution !

6.11.5 Editer listes de paramètres

Colonnes (attributs) affichées / Largeur de colonne :

La liste de paramètres actuellement marquée dans la fenêtre de navigation est représentée dans l'éditeur de tableau selon sa définition dans un fichier descriptif spécifique au système cible, ou selon les réglages standard.

Cela signifie que les valeurs des attributs de chaque paramètre contenu dans la liste sont toutes décrites dans une **ligne**, en fonction de la disposition spécifique au type de liste et de la sélection des colonnes.

Les colonnes peuvent être **développées ou masquées** à titre individuel via le menu contextuel, cela si le curseur se trouve sur la ligne contenant les titres des colonnes.

Afin de modifier la largeur des colonnes, vous pouvez déplacer les lignes de séparation entre les titres de colonnes, et vous disposez en plus, si le pointeur de la souris se trouve sur un titre de colonne, de deux commandes dans le menu contextuel. La **largeur de colonne standard** est calculée de telle sorte que toutes les colonnes puissent être visibles dans la fenêtre. **L'agrandissement de colonne** se rapporte à la colonne actuellement à l'avant-plan et élargit la colonne de telle sorte que chaque entrée soit visible dans son entièreté.

Commandes pour l'édition des entrées de paramètres :

Les commandes suivantes permettant l'édition sont disponibles dans le **menu contextuel** ou dans les menus '**Insérer**' ou '**Extras**':

Insertion / Effacement de lignes :

Insérer ligne resp. **Nouvelle Ligne**	La nouvelle entrée (ligne) est insérée au dessus de l'entrée actuellement à l'avant-plan.
Ligne après resp. **Nouvelle Ligne** après Shortcut:<Strg><Enter>	La nouvelle entrée (ligne) est insérée en dessous de l'entrée actuellement à l'avant-plan.
Effacer ligne	La ligne se trouvant actuellement à l'avant-plan est effacée. Raccourci clavier : <Maj>+<Suppr>
Couper, Copier, **Insérer ligne**	La ligne se trouvant actuellement à l'avant-plan est coupée, insérée ou copiée. L'insertion se fait toujours au dessus de la ligne se trouvant actuellement à l'avant-plan.

Édition des valeurs des attributs :

Lorsqu'une nouvelle ligne est insérée pour une entrée de paramètre, les champs d'attributs sont automatiquement remplis des valeurs par défaut spécifiques à la cible. Afin de saisir ou de modifier une valeur, cliquez avec la souris sur le champ correspondant. Si celui-ci est éditable, vous avez accès à un cadre d'édition. Pour les champs dans lesquels une variable du projet CoDeSys peut être saisie, vous disposez de la liste de sélection pour l'édition (<F2>).

Appuyez sur la touche **<Entrée>** afin de clôturer une saisie.

Les touches directionnelles vous permettent de sauter sur un autre champ.

<Suppr> efface le contenu du champ dans lequel le curseur se trouve.

Pour permuter le format de saisie 'décimal' et 'hexadécimal', utilisez la commande **Format Déc/Hex** dans le menu 'Extras'.

La touche **<F6>** vous permet de passer à la Fenêtre de navigation (et de revenir).

<u>Options :</u>

En dessous de la partie tableau de l'éditeur, il est possible d'activer les options suivantes, selon le type de liste :

Charger avec programme : La liste est automatiquement chargée dans l'automate programmable lors de l'ouverture de session.

Actions synchrones : momentanément sans fonction.

6.11.6 Manager des paramètres en Mode en Ligne

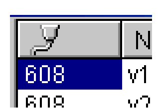
Transfert de listes entre l'éditeur et l'automate programmable

En mode En ligne, il est possible de charger les listes de paramètres créées dans le gestionnaires des paramètres vers l'automate (= **téléchargement**), et de charger celles s'y trouvant dans l'éditeur (= **téléchargement en amont**). La taille maximale des listes pour les types Paramètres et Variables sont définies en fonction du système cible.

Veuillez noter: Lors d'une ouverture de session, un téléchargement est automatiquement effectué pour toutes les listes pour lesquelles l'option '**Charger avec programme**' a été activée dans le gestionnaire des paramètres !

En outre, il est possible **d'écrire** les valeurs individuellement sur l'automate programmable.

Pour l'**affichage des valeurs actuelles** de chaque paramètre, une colonne supplémentaire est disponible en mode En ligne (la première colonne) dans le gestionnaire des paramètres :



Les commandes suivantes sont disponibles dans le **menu 'Extras'**:

Supprimer liste	La liste marquée dans la fenêtre de navigation est supprimée sur l'automate programmable.
Ecrire liste	On accède au dialogue '**Copier objets**', dans lequel on doit sélectionner dans toutes les listes disponibles celles qui doivent être chargées sur l'automate programmable. Le téléchargement s'effectue après confirmation par OK.
Lire liste	Toutes les listes de type 'Paramètre' sont chargées de l'automate programmable sur le gestionnaire des paramètres.
Ecrire des valeurs	Toutes les valeurs dans la colonne 'Valeur' sont écrites sur l'automate dans la liste des paramètres. Pour écrire des valeurs individuelles, double-cliquez sur le champ correspondant de la colonne afin d'avoir accès au dialogue 'Écrire valeurs', comme avec la commande 'En ligne' 'Écrire valeurs'.
Ecrire des valeurs de défault	Les valeurs dans la colonne 'Défaut' sont écrites sur l'automate dans la liste correspondante des paramètres.
Appliquer des valeurs	Les valeurs actuelles des paramètres sont lues à partir de l'automate programmable et écrites dans la colonne 'Valeur'.

La commande **Format Déc/Hex** aussi est disponible en mode en ligne, pour commuter le format d'indication entre decimal et hexadecimal.

Listes de paramètres dans le projet d'initialisation

Le système cible détermine si les contenus du Gestionnaire des paramètres doivent être repris dans un projet d'initialisation lors de la création de ce dernier.

6.11.7 Exporter / Importer listes de paramètres

'Extras' 'Exporter'

La commande 'Exporter' du menu 'Extras' vous permet d'exporter les listes du gestionnaire des paramètres dans un fichier XML pouvant être à nouveau inséré par le biais de la commande 'Extras' 'Importer' (dans un autre projet, par exemple). Pour ce faire, vous avez accès au dialogue standard d'enregistrement de fichier avec extension par défaut ***.prm**. On écrit toujours toutes les listes disponibles dans le gestionnaire des paramètres dans le fichier d'exportation.

Les contenus du gestionnaire des paramètres peuvent également être exportés par le biais de la fonction générale d'exportation ('Projet' Export').

'Extras' 'Importer'

La commande 'Importer' du menu 'Extras' vous permet d'importer le contenu d'un fichier XML qui décrit des listes de paramètres. Ce fichier peut par exemple avoir été créé par le biais de la commande 'Extras' 'Exporter' et dispose donc dans le cas standard de l'extension *.prm.

Si le fichier d'importation contient une liste dont le nom existe déjà dans le gestionnaire des paramètres, vous avez accès à un dialogue vous demandant si la liste existante doit être écrasée.

6.12 Configuration de la Cible

La configuration de la cible se trouve sous l'onglet Ressources en tant qu'objet. On détermine ici sur quel automate (système cible) et selon quelle configuration le projet doit tourner. Suite à la commande **'Projet' 'Nouveau'**, vous êtes directement invité à sélectionner une 'cible', c'est-à-dire une configuration prédéfinie.

La liste de sélection dépend des "Target Support Packages" (**TSP**) installés sur l'ordinateur. Ceux-ci décrivent des configurations de base spécifiques aux plate-formes et déterminent en même temps dans quelle mesure celles-ci peuvent encore être adaptées par l'utilisateur dans les boîtes de dialogue des réglages du système cible.

Veuillez noter : Si aucun TSP n'est disponible, le réglage '**None**' sera affiché dans la liste de sélection des systèmes cibles, ne permettant dès lors aucun réglage et passant automatiquement au mode simulation.

Target-Support-Package

Un Target Support Package (TSP) doit auparavant être installé à l'aide du programme d'installation **InstallTarget** avant de démarrer le programme. Ceci peut être compris dans l'initialisation **CoDeSys**.

Un TSP se compose de tous les fichiers de configuration et d'extension nécessaires à l'utilisation d'un automate précis (système cible, Target) par une application. Sont configurés : le générateur de code, la structure de la mémoire, l'étendue des fonctions de l'automate et les modules E/S. Il faut en outre intégrer les bibliothèques, les pilotes de gateway, les fichiers d'erreur et d'initialisation pour le navigateur d'API, etc. L'élément central d'un TSP se compose d'un ou de plusieurs fichiers cibles. Un **Fichier cible** fait référence aux fichiers supplémentaires nécessités pour la configuration de la cible, mais il peut cependant se les partager avec d'autres fichiers cibles.

En général, un fichier cible porte l'extension ***.trg**, et son format est binaire. Les entrées des configurations sont pourvues de définitions supplémentaires qui déterminent si elles peuvent être visualisées par l'utilisateur dans la boîte de dialogue des réglages du système cible et si elles peuvent être éditées en cet endroit.

Durant l'installation d'un TSP, un fichier cible correspondant est créé pour chaque système cible dans un propre répertoire, et son chemin d'accès est enregistré. Les fichiers s'y rapportant sont également copiés sur l'ordinateur conformément à un **fichier d'info *.tnf** compris dans le TSP. Le nom du répertoire cible coïncide avec le nom de la cible. La classification dans un répertoire portant le nom du fabricant est également proposée.

Les fichiers installés avec un Target Support Package sont lus au démarrage du programme **CoDeSys**. Les réglages du système cible entrepris dans les boîtes de dialogue du système de programmation sont enregistrés avec le projet correspondant.

Veuillez noter : Si un nouveau fichier cible est utilisé ou si le fichier cible existant est modifié, **CoDeSys** doit être redémarré afin de rendre la version actualisée disponible.

Boîte de dialogue de la configuration de la cible

La boîte de dialogue de la **Configuration du système cible** s'ouvre automatiquement dès qu'un nouveau projet est créé. Dans les autres cas, vous pouvez l'obtenir via l'option 'Configuration du système cible' sous l'onglet 'Ressources'.

Sélectionnez sous **Configuration** une des configurations du système cible qui vous sont proposées.

Si aucun Target Support Package n'est installé, la configuration '**None**' est seule proposée, menant automatiquement au mode simulation. Si vous optez pour une des configurations, les possibilités d'adaptation qui vous sont offertes dépendent des entrées dans le fichier cible servant de base. Si vous optez pour une configuration de système cible provenant d'un TSP pour lequel aucune licence valable n'existe sur l'ordinateur, vous êtes invité à choisir une autre cible.

Si une configuration munie de "HideSettings" dans le fichier cible est réglée, seul le nom de la configuration apparaît. Dans les autres cas, quatre onglets sont à disposition permettant l'adaptation ou la représentation de la configuration du système cible :

- **Plate-forme de la cible**
- **Composition** de la mémoire
- **Général**
- **Fonctions de réseau**
- **Visualisation**

Attention : Gardez en mémoire que toute modification de la configuration préréglée du système cible peut avoir de lourdes conséquences sur le comportement du système cible

Après un changement, vous pouvez revenir aux valeurs de la configuration par défaut par le biais du bouton **Par défaut**.

Pour de plus amples informations sur les différents dialogues permettant la configuration du système cible, voir chapitre Appendice G.

6.13 PLC-Browser

Dans le cas du PLC-Browser, il s'agit d'un moniteur de commande (terminal) basé sur du texte. Les commandes permettant l'accès à certaines informations provenant de l'automate sont introduites sur une ligne de saisie et envoyées à l'automate sous forme de chaînes de caractères. La chaîne de caractère de réponse obtenue en retour est visualisée dans une fenêtre de résultats du navigateur. Cette fonction sert à un diagnostic et un débogage.

Les commandes à disposition pour le système cible configuré se composent du set standard de **CoDeSys** en plus d'un set d'extension possible provenant du fabricant de l'automate. Elles sont gérées dans un fichier INI et sont implémentées en conséquence dans le système d'exécution.

6.13.1 Généralités quant à l'usage du PLC-Browser

Sélectionnez dans l'onglet **Ressources** l'entrée **PLC-Browser**. (La disponibilité dépend de la configuration du système cible.)

Le PLC-Browser CoDeSys

```
?
? - show implemented commands
mem     - Memorydump
memc    - Memorydump relative to code-startaddress
memd    - Memorydump relative to data-startaddress
reflect - reflect current command (test!)
dpt     - get data-pointer-table
ppt     - get POU-table
pid     - get project ID
pinf    - get project info
tsk     - get IEC-task-list and IEC-task-infos
Custom help Nr.1
Custom help Nr.2
Custom help Nr.3
```

Le navigateur se compose d'une ligne de saisie des commandes et d'une fenêtre d'affichage / de résultats.

La ligne de saisie propose dans une boîte de sélection une liste de toutes les commandes saisies depuis le début du projet (historique de la saisie). Elles sont à nouveau disponibles jusqu'à ce que le projet soit refermé. Des nouvelles commandes ne sont reprises dans la liste que si elle se différencient de celles déjà présentes.

Grâce à la touche <Entrée>, la commande saisie est envoyée à l'automate. S'il n'y a pas de connexion En ligne, la commande est affichée dans la fenêtre des résultats telle qu'elle a été envoyée à l'automate ; sinon, cette même fenêtre affiche la réponse de l'automate. Si une nouvelle commande est envoyée à l'automate, le contenu de la fenêtre des résultats est effacé.

Les commandes peuvent être saisies sous la forme de chaînes de caractères (voir ci-dessous, Saisie des commandes), mais l'utilisation de macros est également possible (voir ci-dessous, Utilisation de macros).

6.13.2 Saisie des commandes avec le PLC-Browser

Le PLC-Browser met en principe toutes les commandes standard **3S encodées** dans le système d'exécution à disposition. Il s'agit de fonctions permettant une manipulation directe de la mémoire, pour l'affichage de fonctions du projet et de statut et pour l'espionnage du système d'exécution. Elles sont décrites dans un fichier INI pour le navigateur qui fait partie intégrante du "Target Support Package". Ces commandes standard peuvent être complétées par d'autres commandes spéciales, par exemple une fonction diagnostic ou d'autres messages de statut de l'applicatif de l'automate. Une extension de cette liste de commandes doit être exécutée aussi bien dans l'interface du système d'exécution que par des entrées supplémentaires dans le fichier INI du navigateur.

Lors de l'ouverture du projet, **une liste des commandes** disponibles avec le PLC-Browser est générée à partir des entrées du fichier INI de ce même navigateur. Elle peut être appelée en tant que liste de sélection pour l'édition par le biais du bouton ... dans la boîte de dialogue **'Insérer commande standard'** ou de la touche <F2>. Vous pouvez taper la commande ou encore la sélectionner hors de la liste en double-cliquant dessus.

La syntaxe de commande générale est :

<MOT CLÉ><ESPACE><PARAMÈTRES DÉPENDANT DU MOT CLÉ>

Le mot clé est la **commande**. Les **paramètres** servant à son extension sont décrits dans l'info-bulle correspondante de la fenêtre de la liste de sélection pour l'édition.

La commande envoyée est répétée dans la fenêtre des résultats, la réponse de l'automate apparaît en dessous.

Exemple : accès à l'identification du projet de l'automate grâce à la commande "pid" Saisie dans la ligne des commandes :

```
pid.....
```

Affichage dans la fenêtre des résultats :

```
pid
Project-ID: 16#0025CFDA
```

Pour chaque commande standard, vous pouvez afficher un **texte d'aide** par le biais de ?<ESPACE><MOT CLÉ>. Celui-ci sera également défini dans le fichier ini.

Les commandes ci-dessous sont intégrées au système d'exécution et dans le fichier INI avec les entrées correspondantes pour les listes de sélection, les infos-bulles et l'aide :

Commande	Description
?	Le système d'exécution donne une liste des commandes possibles
mem	Image en format hexadécimal d'une partie de la mémoire Syntax 1: mem <Anfangsadresse> <Endadresse> Syntax 2: mem <Anfangsadresse>-<Endadresse> Adressangaben können dezimal, hexadezimal(Prefix 16#) oder als Makro geschrieben werden.
memc	Image en format hexadécimal se rapportant à l'adresse initiale du code de l'automate. Comme avec mem, les données sont ajoutées à la zone de code.
memd	Image en format hexadécimal se rapportant à l'adresse de base des données de l'automate. Comme avec mem, les données sont ajoutées à la zone de données.
reflect	Rendre la ligne actuelle de commande à des fins de tests
dpt	Lire le tableau des pointeurs de données
ppt	Lire le tableau POU
pid	Lire l'identification du projet

pinf	Lire les informations relatives au projet
tsk	Afficher la liste des tâche CEI et les informations s'y rapportant
od	Éditer le dictionnaire d'objets
pod	Éditer la définition des variables réseau
startprg	Démarrer le programme d'automate
stopprg	Arrêter le programme d'automate
resetprg	Réinitialiser le programme d'automate (reset).
resetprgcold	Réinitialiser le programme d'automate à froid (cold reset). Seules les données non rémanentes sont initialisées.
resetprgorg	Réinitialiser le programme d'automate à son état initial (original reset). Le programme utilisateur actuel ainsi que toutes les données (y compris les données rémanentes et persistantes) sont effacés.
reload	Charger à nouveau le projet d'initialisation
getprgprop	Visualiser les propriétés du programme (Nom de projet, Titre, Version, Auteur, Date de la dernière modification)
getprgstat	Visualiser le statut du programme
filecopy	Copier un fichier [de] [vers]
filerename	Renommer un fichier [ancien] [nouveau]
filedelete	Effacer un fichier [nom du fichier]
saveretain	Sauvegarde des variables rémanentes
restoreretain	Chargement des variables rémanentes
setpwd	Définir mot de passe sur l'automate programmable Syntaxe: setpwd <Mot de passe> [<Niveau>] <Niveau > peut être "0" (préréglage) pour accès uniquement au système de programmation ou "1" pour toutes les applications
delpwd	Effacer mot de passe sur l'automate programmable

Veuillez noter : Le premier mot de la commande saisie est interprété comme étant un mot clé. Si "?<SPACE>" se trouve devant un mot-clé (p.ex. "? mem"), le fichier INI sera parcouru à la recherche de ce mot clé pour trouver la section d'aide adéquate. Si un tel mot clé est disponible, rien n'est envoyé à l'automate, et le texte d'aide est affiché dans la fenêtre des résultats.

Si l'automate ne connaît pas le premier mot de la saisie de commande (<MOT CLÉ>), la réponse 'Keyword not found' ' s'affiche dans la fenêtre des résultats.

6.13.3 Utilisation de macros lors de la saisie de commandes dans le PLC-Browser

Si une commande comportant une macro est introduite dans la ligne de commandes, cette dernière est étendue avant d'être envoyée à l'automate. La réponse apparaît dans la fenêtre des résultats également sous forme étendue.

La syntaxe de saisie est : <MOT CLÉ> <Macro>

<MOT CLÉ> est la commande,

Les macros sont :

%P<NOM> Si <NOM> est du type POU, l'expression sera étendue à <POU-Index>, autrement, il n'y a pas de changement.

%V<NOM> S'il <NOM> est un nom de variable, l'expression sera étendue à #<INDEX>:<OFFSET>, autrement, il n'y a pas de changement (le format #<INDEX>:<OFFSET> sera interprété par l'automate comme adresse mémoire)

%T<NOM> Si <NOM> est un nom de variable, l'expression sera étendue à <TYPE DE VARIABLE>, autrement, il n'y a pas de changement.

%S<NOM> S'il <NOM> est un nom de variable, l'expression sera étendue à <SIZEOF(VAR)>, autrement, il n'y a pas de changement.

Le signe % est ignoré s'il est précédé du symbole d'échappement \ (barre oblique inversée - Backslash). Le symbole d'échappement n'est repris comme tel que lorsqu'il est écrit \\.

Exemple :

Saisie dans la ligne des commandes : (image de la mémoire de la variable .testit ?)

```
mem %V.testit
```

Affichage dans la fenêtre des résultats :

```
mem #4:52
03BAAA24 00 00 00 00 CD CD CD CD ....ÍÍÍÍ
```

6.13.4 Autres options du PLC-Browser

Dans le menu **'Extras'** ou dans la barre d'outils du PLC-Browser, vous trouverez les commandes suivantes vous permettant de manipuler la saisie de la commande ou de l'historique de la liste :

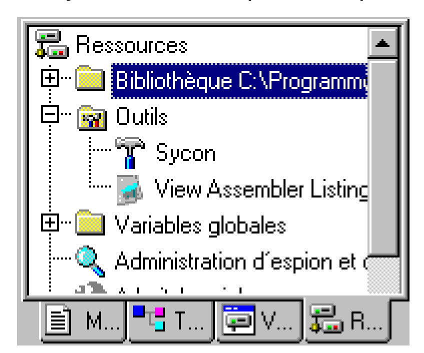
Protocole en sens avant et **Protocole en sens inverse** vous permettent de feuilleter en avant ou en arrière les résultats des demandes. L'enregistrement de l'historique se poursuit jusqu'à ce que vous quittiez le projet.

Annuler la commande vous permet d'interrompre une demande entamée.

Enregistrer la liste du protocole vous permet d'enregistrer les résultats obtenus jusqu'à maintenant dans un fichier texte externe. Cela vous donne accès à la boîte de dialogue 'Enregistrer fichier sous', dans laquelle vous pouvez introduire un nom de fichier avec l'extension ".bhl" (Browser History List). La commande **Imprimer la dernière commande** ouvre la boîte de dialogue standard permettant l'impression. La commande en cours et son résultat dans la fenêtre des messages peuvent être imprimés.

6.14 Outils

L'objet Outils est disponible sous l'onglet Ressources pour autant que ceci ait été configuré en conséquence dans le fichier cible du système cible. Les liens des fichiers exe d'outils externes sont représentés dans 'Outils', et ces outils peuvent être appelés à partir de CoDeSys. Le nombre et le type de liens possibles sont également définis dans le fichier cible. En fonction de cette définition, l'utilisateur peut, dans 'Outils', ajouter ou effacer des liens. La représentation dans l'Organisateur d'objets ressemble par exemple à ceci :



L'exemple donné se compose de quatre liens d'outils : un permet le démarrage de CoDeSys, un autre ouvre le listing d'assembleur dans un éditeur littéral, et les deux dernier permettent l'ouverture de fichiers PDF. Les liens marqués d'un "<R>" ne peuvent plus être modifiés dans le cadre de CoDeSys.

Une utilisation envisageable serait le lien vers un éditeur, par exemple notepad.exe, ou à un fichier PDF précis. Un double-clic sur l'entrée correspondante permettrait alors d'ouvrir le listing assembleur dans le bloc-notes ou encore les fichiers PDF dans Acrobat Reader.

Vous pouvez en outre définir des fichiers qui seraient chargés sur l'automate dès que le lien serait activé.

6.14.1 Caractéristiques des liens existants (Caractéristiques d'objet)

Un clic sur le signe Plus avant l'entrée 'Outils' affiche une liste des liens présents. Lorsqu'un projet est créé, les seuls liens affichés sont ceux qui étaient prédéfinis dans le fichier cible comme entrées fixes. Si vous aviez auparavant travaillé sur le dossier des outils, les liens éventuellement ajoutés par l'utilisateur dans CoDeSys seront également visualisés.

On peut maintenant analyser les Caractéristiques principales des outils ainsi que les Liens individuels:

1. Caractéristiques des 'outils':

Si 'Outils' est marqué dans l'arborescence, on obtient la boîte de dialogue 'Propriétés des outils' dans le menu contextuel ou encore via la commande 'Propriétés' dans le menu 'Projet' 'Objet' :

Tous les outils utilisables dans le cadre de la cible actuelle sont représentés dans le tableau avec les paramètres suivants : La colonne **ID** indique le numéro d'identification univoque d'un outil, viennent ensuite le **Nom du raccourci** tel qu'il apparaît dans l'Organisateur d'objets, ainsi que le nom du fichier exe (**Nom du fichier exécutable)**. Le bouton **'Étendu'** permet d'étendre la boîte de dialogue vers la droite ou de refermer l'extension ouverte :

Boîte de dialogue pour les propriétés des outils

Propriétés des outils

Liste de tous les outils:

ID	Nom du raccourci	Nom du fichier exécutable
0	View Assembler Listing	notepad.exe
1	Sycon	
2	CoDeSys	
4	Tool de Projet $(PROJECT_NAME)	

Répertoire: D:\project2 [Appliquer]

Propriétés:

Numéro fixe: 0

Nom Exe: notepad.exe

Nom de défaut : Tool des Projekts $(PROJECT_NAME) mit viel

Modèle pour les paramètres: $(DISPLAY_NAME)

Charger fichiers modèles: $(INSTANCE_NUMBER);$(PROJECT_DRIVE)\autoexec.bat;$(COMPILE_DIR)\$(PROJECT_NAME).asm;

Editable

[Etendu] [OK] [Annuler]

Après l'extension de la boîte de dialogue, la partie droite reprend les caractéristiques générales du lien tels qu'ils ont été définis dans le fichier cible. Un champ d'édition est en outre disponible, et vous pouvez y définir un **Répertoire** de travail qui sera utilisé pour les actions du fichier Exe. Le chemin d'accès indiqué est enregistré par le biais du bouton **Appliquer**sans que la boîte de dialogue se referme.

'Propriétés des outils :

Numéro fixe: Il s'agit du nombre fixe de liens pour cet outil qui seront automatiquement insérés dans l'Organisateur d'objets. Si "0" est saisi ici, l'utilisateur a la possibilité de créer lui-même un nombre de liens à sa guise. Veuillez noter : pour les liens qui ont été insérés de manière fixe à partir du fichier cible, il n'y a pas que leur nombre qui est défini ; il est également impossible d'en modifier les caractéristiques sous CoDeSys (ces liens sont reconnaissables à la présence d'un "<R>" dans l'Organisateur d'objets).

Nom Exe: Nom de fichier ou chemin d'accès complet du fichier exécutable de l'outil. Le chemin de registre d'un fichier exe peut être saisi ici: "[Chemin de registre].<Entrée renvoyant au fichier exe>". Si aucune entrée n'apparaît, cela signifie que l'extension du fichier indiqué dans les "Paramètres modèles" appelle automatiquement le fichier exe de l'outil concerné par le biais de Windows.

Exemples : "C:\programme\notepad.exe", "345.pdf"

Nom de défault: Nom avec lequel l'outil a été introduit dans l'Organisateur d'objets dans CoDeSys. Le modèle $(INSTANCE NUMBER) peut également être utilisé ici (voir 'Paramètre modèle').

Modèle pour les paramètres:	Modèles pour la définition du fichier qui doit être ouvert dans l'outil. Les modèles suivants peuvent être compris en étant connectés par le signe spécial correspondant : $(PROJECT_NAME) nom du projet en cours (nom du fichier sans l'extension ".pro"). $(PROJECT_PATH) chemin d'accès du répertoire dans lequel le fichier du projet est classé (sans mention du lecteur). $(PROJECT_DRIVE) lecteur sur lequel se trouve le projet en cours. $(COMPILE_DIR) répertoire de compilation du projet (avec mention du lecteur) $(TOOL_EXE_NAME) nom du fichier exe de l'outil. $(DISPLAY_NAME) nom du lien en cours utilisé dans les 'Outils'. $(INSTANCE_NUMBER) numéro du lien (numéro d'instance, numéro courant débutant à "1") $(CODESYS_EXE_DIR) chemin d'accès du répertoire dans lequel le fichier exécutable de CoDeSys se trouve (avec mention du lecteur). Pour la conversion d'un modèle, reportez-vous à la boîte de dialogue des Caractéristiques d'un lien individuel (voir ci-dessous). Exemple : "$(PROJECT_NAME)_$(INSTANCE_NUMBER).cfg" ⇒ le fichier cfg portant le nom <Nom du projet CoDeSys en cours>_<Numéro du lien>.cfg est ouvert dans l'outil
Charger fichiers modèles:	Fichiers, chemins d'accès et modèles pour les fichiers qui doivent être également chargés sur l'automate lors d'un téléchargement. Si l'option **Editable** est activée, la liste de ces fichiers peut être éditée dans la boîte de dialogue des caractéristiques du lien. Si le nom du fichier est indiqué sans chemin d'accès, ce fichier sera recherché dans le répertoire dans lequel le fichier exécutable de CoDeSys se trouve. "a.up;$(PROJECT_NAME).pro;$(INSTANCE_NUMBER).upp" ⇒ les fichiers a.up, <Projet CoDeSys en cours>.pro et <Numéro du lien>.upp seront également chargés sur l'automate lors d'un téléchargement

2. Caractéristiques d'un lien :

Marquez un lien dans l'arborescence des 'outils' et sélectionnez l'option ' Caractéristiques de l'objet' dans le menu contextuel ou dans le menu 'Projet' 'Objet'. Vous avez alors accès à la boîte de dialogue des 'Propriétés de la liaison' contenant les points suivants :

Appel	Appel de l'outil ; chemin d'accès du fichier exe et du fichier défini dans les 'Paramètres modèles' (voir ci-dessus) et affiché sous 'Paramètres'.
Paramètres	Chemin d'accès du fichier accédé à partir de l'outil. Celui-ci résulte de la description de la cible et peut être édité ici pour autant que l'option 'Éditable' soit activée (voir ci-dessous).
Fichiers à charger	Cette liste comprend automatiquement et en premier lieu les **Nom**s **de fichier**s résultants de la description de la cible et qui ont déjà été décrits dans les caractéristiques des outils (voir ci-dessus). Si l'option 'Éditable' est activée (voir ci-dessous la Boîte de dialogue étendue), la liste peut être modifiée ici même. Ouvrez à cet effet la boîte de dialogue '**Indiquer nom de fichier**' par le biais du bouton **Nouveau**; vous pouvez y

introduire un fichier supplémentaire ou un chemin d'accès à un fichier. Si le nom du fichier est indiqué sans chemin d'accès, ce fichier sera recherché dans le répertoire dans lequel le fichier exécutable de CoDeSys se trouve. Le bouton **Effacer** vous permet d'effacer l'entrée actuellement marquée dans la liste.

Le bouton **Standard** ramène les entrées de la boîte de dialogue aux valeurs par défaut indiquées par le fichier cible.

Le bouton **Appliquer** enregistre les réglages entrepris sans pour autant refermer la boîte de dialogue des caractéristiques.

Le bouton **Étendu** permet d'étendre la boîte de dialogue vers la droite, si bien qu'elle ressemble alors à ceci :

Boîte de dialogue pour les caractéristiques des liens

Numéro de raccourci:	Numéro courant, débutant à 1. Tout nouveau lien vers l'outil actuel reçoit chaque fois le numéro directement au dessus. Si vous effacez un lien, les numéros des liens existants seront maintenus. Le numéro de lien peut être utilisé dans d'autres définitions par le biais du modèle $(INSTANCE_NUMBER) (voir par exemple ci-dessus, 'Paramètres Modèles').
ID Outil:	Numéro d'identification univoque de l'outil résultant de sa définition dans le fichier cible.
Nombre de raccourcis de default:	Nombre d'instances d'un outil, correspondant au FixedCount dans le fichier cible. Voir ci-dessus, Caractéristiques des outils.
Editable:	Si cette option est affichée comme étant activée, vous pouvez procéder à des changements dans le champ 'Paramètres' ou dans le tableau des fichiers à charger sur l'automate.

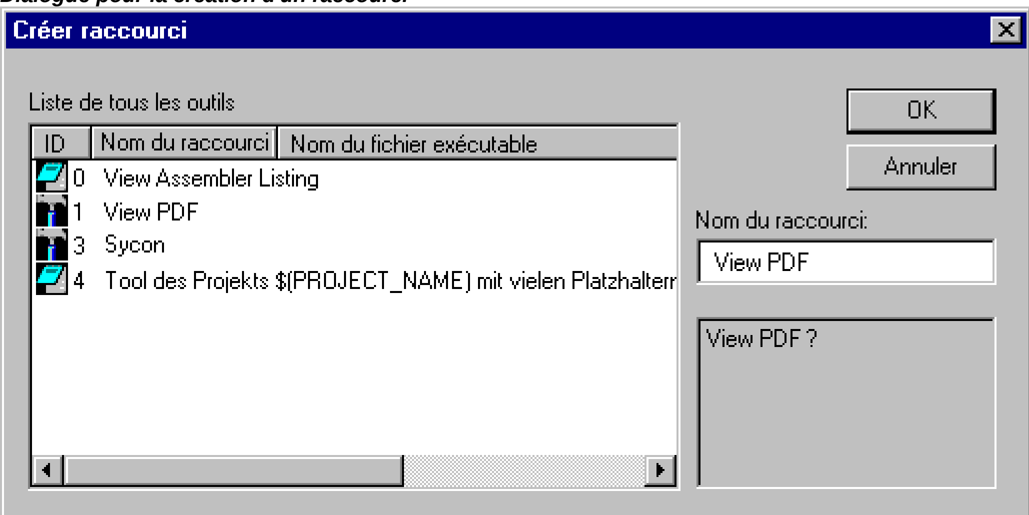
Le bouton **OK** reprend les changements effectués et referme la boîte de dialogue des caractéristiques.

6.14.2 Gestion des liens

Création de nouveaux liens

Si vous avez sélectionné le nœud 'Outils' ou un lien existant dans l'arborescence des ressources, choisissez la commande 'Insérer objet' dans le menu contextuel ou dans le menu 'Projet' 'Objet'. Suite à quoi la boîte de dialogue **'Créer raccourci'** s'ouvre.

Dialogue pour la création d'un raccourci



Tous les outils vers lesquels de nouveaux liens peuvent être établis sont repris dans le tableau. Celui-ci reprend, selon les entrées dans le fichier cible, l'**ID** des outils, le **Nom du raccourci** par défaut et le nom du fichier exe des outils (**Nom du fichier exécutable**).

Pour créer un lien (supplémentaire) avec un des outils proposés, cliquez sur cet outil dans la colonne 'Id'. Le champ **Nom du raccourci** permet de modifier individuellement le nom par défaut de ce nouveau lien que vous confirmez par OK. Ceci n'est possible que lorsqu'un nom n'ayant pas encore été attribué est introduit.

OK referme la boîte de dialogue et le lien qui vient d'être introduit apparaît dans l'arborescence des ressources accompagné de son nom et d'un numéro de lien, qui est d'une unité supérieur au lien auparavant le plus élevé utilisé pour les instances de ce même outil.

La zone en dessous du champ du nom est réservée à des remarques concernant les entrées par l'utilisateur.

Supprimer des liens

Pour supprimer un lien, vous devez sélectionner la commande **Supprimer** dans le menu contextuel (bouton droit de la souris) ou dans le menu 'Projet' 'Objet'. Vous n'avez accès à cette commande que si vous avez sélectionné dans l'arborescence de configuration le lien d'un outil pour lequel aucun nombre fixe de liens n'est spécifié. L'effacement d'une entrée n'entraîne pas de modification dans les numéros des autres liens.

Exécution de liens

Un lien est exécuté par un double-clic sur l'entrée correspondante dans l'arborescence des ressources, ou si la commande 'Editer objet' dans le menu 'Projet' 'Objet' ou dans le menu contextuel (bouton droit de la souris) est exécutée.

Si l'exécution du fichier indiqué par les paramètres n'est pas possible, un message d'erreur adéquat s'affiche. Si un fichier de paramètres n'est pas trouvé, le fichier exe de l'outil est exécuté et une boîte de dialogue apparaît vous demandant si le fichier doit être créé.

Au cas où le fichier exe de l'outil n'est pas trouvé dans le chemin d'accès ou si aucun chemin d'accès n'a été donné, une boîte de dialogue permettant de sélectionner un fichier s'ouvre, incitant l'utilisateur à indiquer le chemin d'accès du fichier exe. Ce chemin sera enregistré à la fermeture de la boîte via OK et sera dans la suite disponible pour cet outil même dans le cadre d'autres projets.

Enregistrement de liens

Lors de l'enregistrement du projet, l'ensemble du nœud 'Outils' est enregistré dans l'arborescence des ressources.

Veuillez noter : Si un projet est enregistré sous un nouveau nom via 'Enregistrer sous', veillez à respecter les points suivants lors de l'utilisation de modèles $(PROJECT_NAME) dans la définition du fichier des paramètres et des fichiers qui devront être chargés sur l'automate :
Pour les liens (FixedCount=0) qui ont été introduits par l'utilisateur dans l'ancien projet, les noms de fichiers doivent être renommés manuellement dans le nouveau projet conformément au nouveau nom de projet. Par contre, le modèle pour un outil dont un nombre fixe de liens a été défini est toujours automatiquement interprété avec le nom actuel du projet.

6.14.3 Les questions les plus importantes en matière d'outils

Pourquoi n'ai-je pas d'entrée 'Outils' sous l'onglet 'Ressources' ?

Ce n'est que si la définition du système cible le prévoit (fichier cible) que vous obtenez l'option 'Outils' sous l'onglet Ressources dans CoDeSys.

Quels outils disposent déjà de liens et quels liens puis-je encore créer ?

Ouvrer le nœud 'Outils' sous l'onglet 'Ressources' grâce à un double-clic sur le signe Plus. Vous pouvez voir quels outils sont déjà rattachés au projet en cours. Si vous avez créé un nouveau projet et que vous n'avez pas encore effectué de changements au sein des outils, il ne s'agira alors que des outils qui ont déjà été spécifiés de manière fixe dans le fichier cible. Dans les autres cas, vous obtenez une liste d'outils qui sera éventuellement déjà adaptée au projet. Pour vérifier si cette liste est encore extensible, sélectionnez la commande 'Insérer objet'. Vous obtenez alors une boîte de dialogue comprenant tous les outils pour lesquels vous pouvez réaliser des liens supplémentaires.

Quelles caractéristiques générales les outils disponibles ont-ils ?

Marquez l'entrée 'Outils' dans l'Organisateur d'objets et sélectionnez la commande 'Caractéristiques des objets' dans le menu contextuel via le bouton droit de la souris. Étendez la boîte de dialogue qui s'est ouverte vers la droite au moyen du bouton 'Étendre'. Vous avez alors la liste des outils disponibles à gauche et les paramètres correspondants à droite. Marquez maintenant un outil particulier au moyen d'un clic sur le symbole d'identité à l'extrême gauche de façon à visualiser dans le champ FixedCount, par exemple, à combien de liens l'outil est limité, quels fichiers ont été chargés sur l'automate lors de l'activation du lien, etc. Les données sont le cas échéant affichées dans des modèles contenant l'interprétation individuelle de chaque lien comme décrit dans le point suivant :

Quelles caractéristiques individuelles les liens déjà disponibles ont-ils ?

Marquez une des entrées sous 'Outils' dans l'Organisateur d'objets et sélectionnez la commande 'Caractéristiques des objets' dans le menu contextuel via le bouton droit de la souris. Si vous appuyez sur le bouton 'Étendu', vous obtenez les paramètres du lien souhaité qui correspondent en partie aux caractéristiques générales d'outils déjà décrites ci-dessus. S'ils ont été définis comme étant 'éditables' dans le fichier cible, vous pouvez procéder ici à la modification de ces paramètres.

Comment puis-je établir un ou plusieurs liens vers un outil ?

Marquez l'entrée 'Outils' dans l'Organisateur d'objets et sélectionnez la commande 'Insérer objet' dans le menu contextuel. Vous obtenez à nouveau une liste des outils disponibles, mais uniquement de ceux dont le nombre maximal d'utilisation (FixedCount) n'a pas encore été atteint. Sélectionnez-en un et appuyez sur OK. L'outil est ensuite visualisé dans l'Organisateur d'objets. Si vous essayez de le rattacher encore une fois, cela ne marchera que si vous lui attribuez un nom modifié, c.-à-d. si vous caractérisez la nouvelle entrée comme étant une nouvelle instance du même outil. Par exemple, les instances de l'outil Toolxy peuvent être nommées Toolxy_1, Toolxy_2 etc.

Comment puis-je modifier les paramètres d’un outil ?

Pour modifier les paramètres d’une instance d’outil, marquez le lien dans l’Organisateur d’objets et sélectionnez la commande 'Caractéristiques des objets' dans le menu contextuel. Dans quelle mesure vous pouvez éditer les paramètres dans les champs textuels dépend de la préconfiguration des outils dans le fichier cible (vérifiez dans la boîte de dialogue étendue si l’option 'Éditable' est activée). Le bouton 'Standard' vous permet toujours de revenir à la configuration par défaut.

Comment puis-je exécuter un lien vers un outil ?

Double-cliquez sur l’entrée du lien dans l’Organisateur d’objets ou choisissez la commande 'Traiter l’objet' dans le menu contextuel ou dans le menu du projet alors que l’entrée est marquée.

7 ENI

7.1 Qu'est-ce que ENI

L'interface **ENI** ('Engineering Interface') permet l'accès du système de programmation à une base de données de projet externe dans laquelle les données résultant de la création d'un projet d'automatisation sont gérées. L'utilisation d'une base de données externe garantit la consistance des données qui peuvent alors être utilisées de manière commune par plusieurs utilisateurs, projets ou programmes, et permet les extensions suivantes dans les fonctionnalités CoDeSys :

- **Gestion de version** pour les projets CoDeSys et les ressources s'y rapportant (objets utilisés en commun) : Si un objet a subi un check-out, été modifié et subi un check-in, une nouvelle version de l'objet est créée dans la base de données, mais les anciennes versions sont cependant gardées et peuvent toujours être à nouveau appelées si besoin est. L'historique des modifications est tracé pour chaque objet et pour un projet global. Les versions peuvent être comparées à la recherche de différences. (Ceci ne vaut pas si un système de fichiers local est utilisé comme base de données.)
- **Fonctionnement multi-utilisateurs** : La version la plus récente d'un ensemble de modules, par exemple les modules d'un projet, peut être rendue accessible à un groupe d'utilisateurs. Les modules ayant subi un check-out par un utilisateur sont marqués auprès des autres utilisateurs comme étant 'en traitement' et ne peuvent être modifiés. Ainsi, plusieurs utilisateurs peuvent travailler au même projet sans que les versions des objets ne s'écrasent mutuellement.
- **Accès via des programmes externes** : En plus du système de programmation CoDeSys, d'autres outils disposant également de l'interface ENI peuvent avoir accès à la même base de données. Il s'agit par exemple de visualisations externes, de systèmes ECAD, etc. qui ont besoin des données créées au sein de CoDeSys ou qui créent eux-mêmes des données.

Afin de créer la base de données sur un autre ordinateur et par cela permettre le fonctionnement multi-utilisateurs, l'interface ENI se compose d'un client et d'une partie serveur. Le système de programmation CoDeSys est également un client du **Processus indépendant du serveur ENI** au même titre que toute autre application qui aurait besoin de l'accès à la base de données. Pour l'installation, la configuration et l'utilisation du serveur ENI, reportez-vous à la documentation correspondante.

L'interface ENI supporte actuellement les bases de données 'Visual SourceSafe 6.0', 'MKS Source Integrity', 'PVCS Version Manager' à partir de V7.5 et un système local de fichiers. Les objets peuvent y être répertoriés dans différents 'Répertoires' (Catégories avec différentes caractéristiques d'accès), subir un check-out pour être traités et être ainsi bloqués aux autres utilisateurs. Les objets peuvent être appelés hors de la base de données dans leur état actuel. En même temps, des objets peuvent n'être gardés que localement, c.-à-d. dans le projet. Le fichier *.pro est la copie de travail d'un projet géré dans la base de données.

7.2 Conditions pour travailler avec une base de données de projet ENI

Pour pouvoir utiliser l'interface ENI dans le système de programmation CoDeSys à des fins de gestion des objets de projet dans une base de données externe, vous devez remplir les conditions suivantes :

Veuillez noter : Pour l'installation, la configuration et l'utilisation du serveur ENI de 3S – Smart Software Solutions GmbH, reportez-vous à la documentation correspondante. Notez également la possibilité d'utiliser l'Explorateur ENI qui permet, combiné au serveur ENI, de gérer la base de données connectée à ce dernier indépendamment du système de base de données utilisé.

- **TCP/IP** doit être disponible pour la connexion entre CoDeSys et le serveur ENI puisque ce dernier utilise le protocole HTTP.

- Un **serveur ENI** doit être installé et démarré localement ou sur un autre ordinateur (*Suite de Serveur ENI*). Un licence valable est obligatoire pour rendre le pilote de base de données disponible. Sans licence, vous ne pouvez utiliser que le pilote du système local de fichiers.
- Configurez les points suivants dans l'administration du serveur ENI, ***ENI Admin*** :
  - L'utilisateur doit être enregistré comme utilisateur possédant les droits d'accès (User Management)
  - Les droits d'accès aux répertoires de la base de données doivent être correctement définis (Access Rights)
  - Recommandation : le mot de passe de l'administrateur permettant l'accès aux programmes *ENI Admin* et *ENI Control* doit être défini immédiatement après l'installation.
- La connexion vers la base de données souhaitée doit être correctement configurée dans le programme de contrôle ***ENI Control*** (Database).
- Une **base de données** de projet pour laquelle il y a un pilote supporté par le serveur ENI doit être installée ; nous recommandons d'installer celle-ci sur l'ordinateur sur lequel le serveur ENI fonctionne. Vous pouvez également utiliser un système local de fichiers pour lequel il y a pour chaque cas un pilote pour serveur ENI.
- Le cas échéant, l'utilisateur (comme Client) ainsi que le serveur ENI doivent être enregistrés tous deux dans l'administration de la base de données comme utilisateurs ayant des droits d'accès. Ceci vaut dans tous les cas pour l'utilisation de SourceSafe comme base de données; pour les autres pilotes de base de données, reportez-vous à la documentation correspondante pour la configuration obligatoire des utilisateurs.
- **L'interface ENI** doit être activée pour le projet en cours (reportez-vous pour ce faire à la boîte de dialogue CoDeSys 'Projet' 'Options' 'Base de données du projet'.
- Vous devez procéder à la **configuration** de la liaison à la base de données pour le projet CoDeSys en cours (reportez-vous pour ce faire aux boîtes de dialogue CoDeSys sous 'Projet' 'Options' 'Base de données du projet).
- L'utilisateur doit **ouvrir une session** (nom d'utilisateur et mot passe) auprès du serveur ENI dans le projet en cours et grâce à la boîte de dialogue d'ouverture de session, obtenue via la commande 'Projet' 'Liaison avec la base de données' 'Login' ou en essayant d'ouvrir une base de données.
- (Reportez-vous également aux instructions abrégées dans le document 'Serveur ENI – Aperçu et démarrage')

7.3 Travailler dans CoDeSys avec la base de données de projet

Les **commandes de la base de données** (appel, check-out, check-in, historique des versions, étiquetage, etc.) permettant de gérer les modules du projet dans la base de donnes du projet ENI sont à disposition dans le projet CoDeSys en cours dès que la liaison à la base de données a été activée et correctement configurée. Reportez-vous également aux 'Conditions pour travailler avec une base de données de projet ENI'. Vous trouverez ces commandes au menu 'Liaison à la base de données'. Ce dernier est un des sous-menus du menu 'Projet' ou se trouve dans le menu contextuel d'un objet individuel marqué dans l'Organisateur d'objets.

L'attribution actuelle d'un objet à une Catégorie de bases de données est affichée à la catégorie Propriétés des objets et vous pouvez la modifier.

Les **Caractéristiques des catégories de bases de données** (paramètres de connexion, droits d'accès, comportement au check-out ou au check-in) peuvent être modifiées dans les boîtes de dialogue des Options de la base de données du projet.

7.4 Catégories au sein de la base de données du projet

On distingue quatre groupes d'objets dans un projet CoDeSys :

L'Interface ENI distingue trois catégories d'objets (catégories d'objets ENI) qui sont gérées dans le système de données : les objets de projet, les objets communs, les objets de compilation.

Un objet peut également appartenir à la catégorie 'Local' s'il n'est pas repris dans la base de données mais s'il est simplement enregistré de manière classique avec le projet.

Dès lors, un module CoDeSys peut être attribué dans le système de programmation à une des catégories Objets du projet, Objets communs ou Local ; les données de compilation n'existent pas encore au sein du projet comme des objet attribuables. L'attribution d'un objet à une des catégories s'effectue automatiquement lors de sa création, en fonction de la configuration des Options de la base de données de projet, mais peut également être modifiée dans la boîte de dialogue des Propriétés des objets.

Chaque catégorie d'objet ENI est configurée séparément dans la boîte de dialogue Configuration ENI (Options de projet, Catégorie base de données de projet). Ceci signifie que chaque catégorie obtient ses propres paramètres relatifs à la connexion la base de données (répertoire, port, nom d'utilisateur, droits d'accès, etc.) et au comportement en cas d'appel, de check-out ou de check-in. Cette configuration vaut alors pour tous les objets qui appartiennent à cette catégorie. Même les données d'accès (nom d'utilisateur, mot de passe) lors de la connexion à la base de données sont à introduire en conséquence pour chaque catégorie. Vous disposez pour cela de la Boîte de dialogue d'accès au système.

Vous avez la possibilité de créer dans chaque base de données un répertoire propre pour chaque catégorie d'objets, mais il est également possible de garder les objets de toutes les catégories dans le même répertoire, vu que l'attribution à une catégorie est une caractéristiques de l'objet et non du répertoire.

Vous trouverez ci-dessous les trois catégories possibles d'objets ENI :

Projet :	pour les objets qui contiennent des informations sources spécifiques au projet, p.ex. modules utilisés en commun au sein d'un projet, important pour un fonctionnement multi-utilisateurs. La commande 'Versions dernières' de CoDeSys permet de reprendre automatiquement dans le projet local tous les objets de cette catégorie provenant du répertoire de projet de la base de données, y compris ceux qui n'y avaient pas encore été créés.
Objets commun :	pour des objets généraux indépendants des projets, par exemple des bibliothèques de modules qui sont normalement utilisés par plusieurs utilisateurs dans différents projets. Attention : la commande 'Versions dernières' de CoDeSys permet de ne copier dans le projet local que les objets de cette catégorie provenant du répertoire de projet de la base de données, qui y sont déjà créés.
Fichiers de compilation :	pour les informations de compilation spécifiques au projet, créées automatiquement par CoDeSys (p.ex. fichiers de symboles), et qui sont également utilisées par d'autres outils. Par exemple, une visualisation requiert les variables d'un système de programmation y compris les adresses, qui ne sont attribuées que lors de la compilation.

Vous pouvez également retirer selon votre gré des modules de projet hors de la gestion de la base de données du projet et les enregistrer exclusivement localement et de manière classique avec le projet.

Interface DDE

CoDeSys dispose d'une Interface DDE (dynamic data exchange). Grâce à cela, **CoDeSys** donne accès aux contenus de variables d'automate programmable et d'adresses CEI à partir d'autres applications disposant également d'une interface DDE.

Lors de l'utilisation de la Gateway DDE, les valeurs de variables peuvent être lues indépendamment du système de programmation **CoDeSys** de l'automate et également représentées dans des applications qui disposent également d'une interface DDE.

Attention : Les adresses directes ne peuvent être lues via le serveur DDE ! Dans ce cas, les variables doivent être créées dans **CoDeSys** avec l'affectation correcte de l'adresse (AT).

Attention : L'interface DDE a été testée avec Word 97 et Excel 97 sous Windows NT 4.0. En cas de problèmes dans la communication DDE, causés par l'utilisation d'autres versions ou par des programmes installés en sus sur votre ordinateur, 3S – Smart Software Solutions ne peut être tenue pour responsable.

7.5 Interface DDE du système de programmation CoDeSys

Activation de l'interface DDE

L'interface DDE est activée dès que l'on accède à l'automate programmable (ou à la simulation).

Accès standard aux données

Une interrogation DDE est composée de trois parties:

1. Le nom du programme (dans le cas présent: **CoDeSys**),
2. Le nom du fichier et
3. Le nom de variable devant être lu.

Nom du programme: **CoDeSys**

Nom de fichier: chemin d'accès complet du projet à partir duquel les données doivent être extraites (C:\beispiel\bsp.pro).

Nom de variable: nom de la variable tel qu'il est spécifié dans le gestionnaire d'espion et de recettes. (z.B. PLC_PRG.TEST)

Quelles sont les variables qui peuvent être lues ?

Toutes les adresses et variables peuvent être lues. L'entrée des variables ou des adresses s'effectue conformément aux entrées dans le gestionnaire d'espion et de recettes.

Exemples:

%IX1.4.1	(* Lit l'entrée 1.4.1 *)
PLC_PRG.TEST	(* Lit la variable TEST du module PLC_PRG *)
.GlobVar1	(* Lit la variable globale GlobVar1 *)

Lier des variables à WORD

Pour obtenir dans Microsoft WORD la valeur actuelle de la variable TEST du module PLC_PRG, via l'interface DDE, il faut entrer dans WORD un champ quelconque ('Insérer' 'Champ'), par exemple la date. En cliquant à présent avec la touche droite de la souris sur le champ, et en sélectionnant la commande 'Afficher fonction champ', vous pouvez modifier la fonction champ dans le texte de votre choix. Si nous appliquons cela à notre exemple, nous obtenons ceci:

{ DDEAUTO CODESYS "C:\\CODESYS\\PROJECT\\IFMBSP.PRO" "PLC_PRG.TEST" }

Cliquez une nouvelle fois avec la touche droite de la souris sur le champ et donnez la commande 'Actualiser champ'. Le contenu de la variable recherché apparaît dans le texte.

Lier des variables à EXCEL

Pour affecter une variable à une cellule dans Microsoft EXCEL, il faut entrer ceci dans EXCEL:

=CODESYS|'C:\CODESYS\PROJECT\IFMBSP.PRO'!PLC_PRG.TEST

En effectuant 'Editer' 'Liens' pour ce lien, on obtient:

Type: CODESYS

Fichier source: C:\CODESYS\PROJECT\IFMBSP.PRO

Elément: PLC_PRG.TEST

Accéder à des variables via Intouch

Convenez pour votre projet d'un nom d'accès DDE <AccessName> comportant le nom d'application CODESYS et le topic name DDE C:\CODESYS\PROJECT\IFMBSP.PRO.

Vous pouvez à présent convenir de variables de type DDE portant le nom d'accès <AccessName>. En ce qui concerne l'item name, il faut à nouveau entrer le nom de la variable (p.ex. PLC_PRG.TEST).

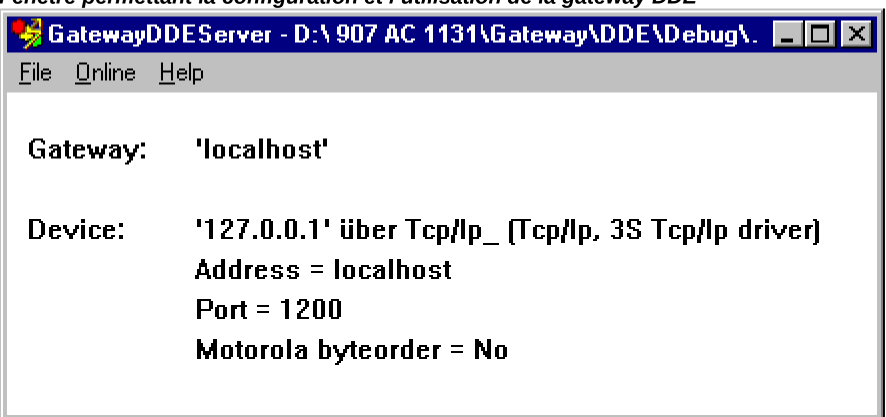
7.6 Communication DDE via la gateway DDE

Utilisation de la Gateway DDE

La gateway DDE peut utiliser les symboles créés au sein du projet CoDeSys (voir 'Projet' 'Options' 'Configuration des symboles') pour communiquer avec d'autres clients ou avec l'automate. Elle peut répondre aux requêtes venant par les interfaces DDE d'applications comme par exemple Excel. Ainsi, les valeurs de variables provenant de l'automate peuvent être visualisées dans d'autres applications.

Dès que vous démarrez la gateway DDE, une fenêtre s'ouvre vous permettant de configurer les paramètres de démarrage et de connexion. Pour ce faire, vous pouvez appeler un fichier de configuration existant ou encore redéfinir les paramètres.

Fenêtre permettant la configuration et l'utilisation de la gateway DDE



La commande **'Fichier' 'Ouvrir'** vous permet d'appeler une configuration déjà enregistrée dans un fichier. La boîte de dialogue standard pour la sélection d'un fichier s'ouvre. La recherche porte par défaut sur des fichiers pourvus de l'extension ".cfg". Lorsqu'un fichier de configuration a été sélectionné, les paramètres de configuration pour la gateway et pour l'appareil cible (Device) s'affichent.

Si l'option **'Fichier' 'Autochargement'** est activée, la gateway DDE s'ouvre automatiquement configurée telle qu'elle était lors de sa dernière utilisation.

Si la gateway est démarrée sans configuration et sans le réglage Autochargement, seules les notions 'Gateway:' et 'Appareil:' sont affichés dans la fenêtre. Vous devez alors procéder à la configuration.

La commande **'Fichier' 'Settings'** donne accès à la boîte de dialogue '**Settings Gateway**' dans laquelle vous pouvez définir les paramètres suivants :

Boîte de dialogue pour la définition des paramètres de la gateway DDE

Server settings

Motorola byteorder | OK
Check identity | Cancel
Updaterate (ms): 100
Timeout (ms): 2000
Tries: 1

Pour configurer la connexion à la gateway actuelle, appelez la boîte de dialogue **'Paramètres de communication'** via la commande 'En Ligne' 'Paramètres'. Vous obtenez la même boîte de dialogue que dans CoDeSys sous 'En Ligne' 'Paramètres de communication'. La configuration doit coïncider avec celle qui a été effectuée dans le projet CoDeSys correspondant.

Vous pouvez enregistrer la configuration actuelle de la gateway DDE dans un fichier à l'aide de la commande **'Fichier' 'Sauvegarder'**. À cet effet, la boîte de dialogue standard permettant de sauvegarder un fichier s'affiche, et l'extension ".cfg" y est attribuée par défaut.

Si la gateway doit être activée, vous devez accéder au système par le biais de la commande **'En Ligne' 'Accès au système'** (suite à quoi le symbole de la gateway s'allume dans la barre d'état). La connexion se met en place telle qu'elle a été configurée et vous pouvez utiliser les symboles correspondants. Veuillez noter que ceux-ci doivent tout d'abord être créés dans le projet CoDeSys.

La commande **'En Ligne' Quitter le système'** vous permet de quitter le système.

Accès standard aux données

Une interrogation DDE est composée de trois parties:

1. Le nom du programme (dans le cas présent: CoDeSys),

2. Le nom du fichier et

3. Le nom de variable devant être lu.

Nom du programme: GatewayDDEServer

Nom de fichier: chemin d'accès complet du projet à partir duquel les données doivent être extraites (C:\beispiel\bsp.pro).

Nom de variable: nom de la variable tel qu'il est spécifié dans le gestionnaire d'espion et de recettes. (par exemple : PLC_PRG.TEST)

Quelles sont les variables qui peuvent être lues ?

Toutes les adresses et variables peuvent être lues. L'entrée des variables ou des adresses s'effectue conformément aux entrées dans le gestionnaire d'espion et de recettes.

Exemples:

%IX1.4.1	(* Lit l'entrée 1.4.1 *)
PLC_PRG.TEST	(* Lit la variable TEST du module PLC_PRG *)
.GlobVar1	(* Lit la variable globale GlobVar1 *)

Lier des variables à WORD via la gateway DDE

Saisissez un champ ('Insérer' 'Champ') selon votre gré dans WORD, par exemple la date. En cliquant à présent avec la touche droite de la souris sur le champ, et en sélectionnant la commande "Afficher

codes de champs", vous pouvez éditer le texte du code de champs : Pour accéder à la variable TEST du module PLC_PRG dans le projet BSP.PRO, introduisez ceci :

{ DDEAUTO GATEWAYDDESERVER "BSP.PRO" "PLC_PRG.TEST" }

Cliquez à nouveau avec la touche droite de la souris sur le champ et donnez la commande "Actualiser champ". Le contenu de la variable recherché apparaît dans le texte.

Lier des variables à EXCEL via la gateway DDE

Conformément à la procédure décrite plus haut, l'expression suivante est introduite dans la cellule qui représente la valeur de variable correspondante :

=GATEWAYDDESERVER|<nom de fichier>!<nom de variable>

Exemple :

```
=GATEWAYDDESERVER|'bsp.pro'!'PLC_PRG.TEST'
```

Si le champ est mis à jour, le contenu de la variable s'affiche.

En effectuant "Editer" + "Liens" pour ce lien, on obtient:

Type : GATEWAYDDESERVER

Fichier source : BSP.PRO

Elément : PLC_PRG.TEST

Options de lignes de commande pour la gateway DDE

Si la gateway DDE est démarrée par le biais d'une ligne de commande, vous disposez des options suivantes :

/n	La boîte de dialogue d'info n'apparaît pas automatiquement		
/s	Affichage de la fenêtre de la boîte de dialogue	/s=h /s=i /s=m /s=n	aucune réduite (icone) agrandie normale
/c	fichier de configuration qui doit être chargé automatiquement	/c=<fichier de config>	
/o	on accède au mode En ligne avec la configuration choisie (autochargement ou fichier donné par "/c=")		

Exemple :

Saisie dans la ligne des commandes :

```
GATEWAYDDE /s=i /c="D:\DDE\conf_1.cfg"
```

La gateway DDE démarre et la fenêtre de la boîte de dialogue apparaît sous la forme d'une icone, la configuration du serveur enregistrée dans le fichier conf_1.cfg est chargée.

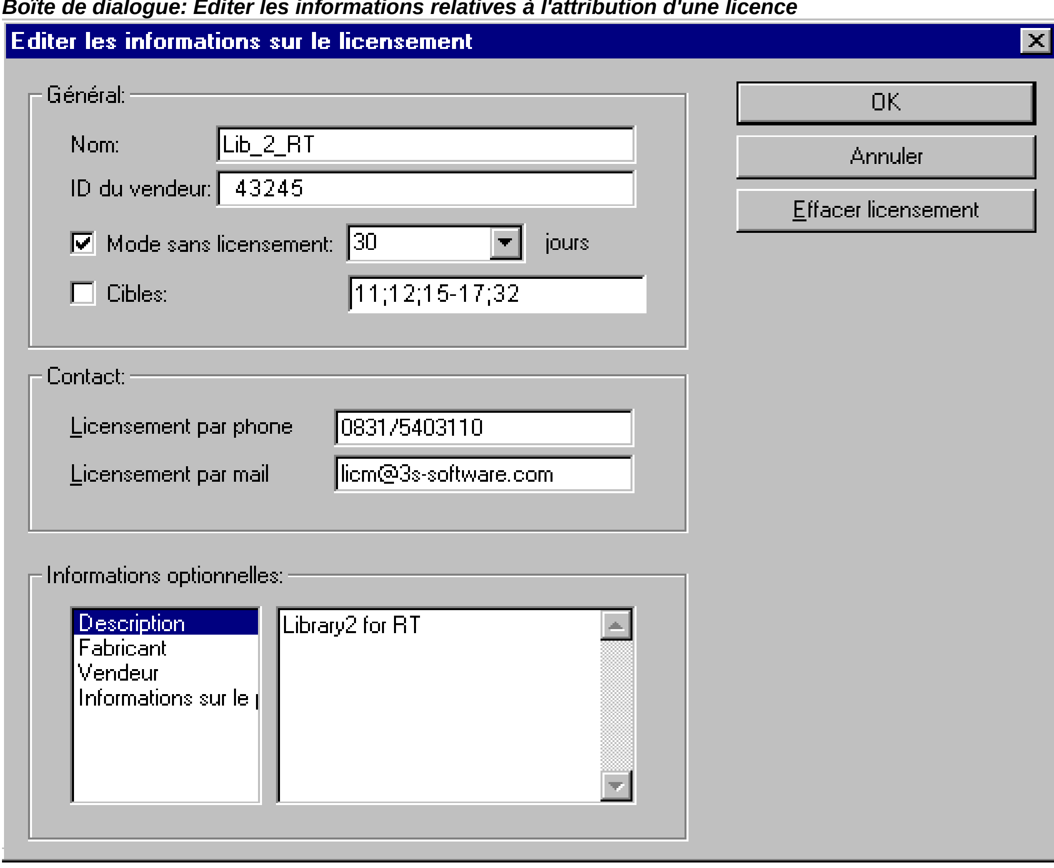
8 L'attribution de licences dans CoDeSys

3S Licensing Manager est un outil permettant une gestion aisée des licences des modules 3S sur l'ordinateur local. Dans CoDeSys, des projets peuvent être créés et enregistrés comme bibliothèques soumises à licence. Lors de l'installation d'un module 3S soumis à licence, on installe également le gestionnaire de licences.

8.1 Création d'une bibliothèque soumise à licence

Si un projet CoDeSys doit être enregistré comme bibliothèque protégée par une licence, alors en utilisant la commande 'Fichier' 'Enregistrer sous...', vous pouvez, par le biais de la boîte de dialogue **Editer les informations sur l'attribution d'une licence** saisir les Informations relatives à la licence. Ces informations seront reprises dans les informations du projet et pourront être visualisées lors de l'Utilisation de la bibliothèque dans les caractéristiques des objets par le gestionnaire de bibliothèques.

Boîte de dialogue: Editer les informations relatives à l'attribution d'une licence



<u>Général :</u>

Nom : il s'agit du nom du module de bibliothèque sous lequel il est géré au sein du gestionnaire de licences. Cette entrée est obligatoire.

ID du vendeur:identification du fabricant, en fonction du programme de gestion des licences spécifiques aux fabricants.

Mode sans licence : activez cette option si le module peut être utilisé sans licence (mode démonstration) et introduisez le nombre de jours après lequel cette licence de démonstration arrive à expiration. Le nombre de jours est toujours automatiquement arrondi à la dizaine supérieure par le gestionnaire des licences (10, 20, 30, ...). Si vous ne saisissez aucun nombre, le module peut être utilisé sans limite dans le temps.

Cibles : introduisez ici l'identité du ou des systèmes cible pour lesquels la licence est valide. Plusieurs ID peuvent être données dans une liste, séparées par point-virgule, ou comme zone. Exemple : "12;15-19;21"

Contact :

Licensement par téléphone: / par e-mail: Introduisez ici le numéro de téléphone ou l'adresse e-mail auxquels l'utilisateur peut demander une licence. Ces entrées sont obligatoires.

Informations optionnelles :

La fenêtre de droite est destinée au texte que vous pouvez saisir selon votre gré en regard du point marqué dans la fenêtre de gauche :

Description, Fabricant, Vendeur, Informations sur le prix

Veuillez noter :

- Une bibliothèque sauvegardée avec des informations relatives à une licence devrait aussi être munie d'un mot de passe. Si vous souhaitez sauvegarder le fichier sans mot de passe, vous en êtes prévenu par un message.
- Pour les bibliothèques 3S, ce n'est pas nécessaire de garder les infos relatives à la licence dans un fichier descriptif de module séparé, car elles sont enregistrées de manière interne et automatiquement stockées sur l'ordinateur lors de l'utilisation de la bibliothèque. Pour les autres, et également par exemple les modules externes (pas 3S), un fichier descriptif doit être disponible en format XML et pouvant être lu par le gestionnaire de licences 3S.

Reportez-vous à cet effet à la documentation du *3S LIcensing Manager*.

9 APPENDICE

Appendice A Les opérateurs CEI et fonctions supplémentaires d'extension des normes

CoDeSys supporte tous les opérateurs CEI. Ceux-ci sont connus implicitement dans la totalité du projet, contrairement aux fonctions standard (bibliothèque standard). Outre les opérateurs CEI, CoDeSys supporte les opérateurs suivants non exigés par la norme : INDEXOF et SIZEOF (voir opérateurs arithmétiques), ADR et BITADR (voir opérateurs d'adressage).

Attention: Si des types de données avec virgules flottantes sont utilisées, le résultat d'une calculation dépendre de la system cible choisi !

Dans les modules du projet les opérateurs sont utilisées comme fonctions.

9.1.1 Opérateurs arithmétiques

ADD

Addition de variables de type BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, REAL et LREAL.

On peut également additionner deux variables TIME, le résultat sera toujours un temps (p.ex. t#45s + t#50s = t#1m35s)

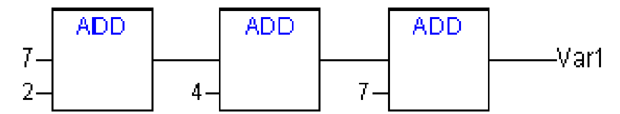
Exemple en langage IL:

```
LD 7
ADD 2,4,7
ST  var1
```

Exemple en langage ST:

```
var1 := 7+2+4+7;
```

Exemple en langage FBD:



MUL

Multiplication de variables du type BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, REAL et LREAL.

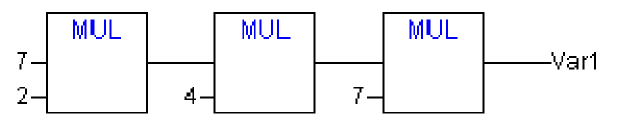
Exemple en langage IL:

```
LD 7
MUL 2,4,7
ST  var1
```

Exemple en langage ST:

```
var1 := 7*2*4*7;
```

Exemple en langage FBD:



SUB

Soustraction d'une variable de type BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, REAL et LREAL d'une autre variable d'un de ces types.

Une variable TIME peut aussi être soustraite d'une autre variable TIME, et le résultat est de nouveau de type TIME. Tenez compte du fait que les valeurs TIME négatives ne sont pas définies.

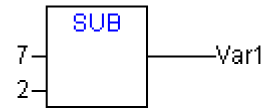
Exemple en langage IL:

```
LD 7
SUB 8
ST  var1
```

Exemple en langage ST:

```
var1 := 7-2;
```

Exemple en langage FBD:



DIV

arithmétique:Division d'une variable de type BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, REAL et LREAL par une autre variable d'un de ces types.

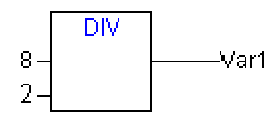
Exemple en langage IL:

```
LD 8
DIV 2
ST  var1
```

Exemple en langage ST:

```
var1 := 8/2;
```

Exemple en langage FBD:



Si dans votre projet vous définissez des fonctions des **CheckDivByte**, **CheckDivWord**, **CheckDivDWord** et **CheckDivReal** ,vous pourrez vérifier la valeur du diviseur lors de l'utilisation de l'opérateur DIV, par exemple pour éviter une division par zéro. Le nom de la fonction est fixé et ne peut être modifié.

Exemple de mise en œuvre de la fonction CheckDivReal :

```
FUNCTION CheckDivReal : REAL
VAR_INPUT
 divisor:REAL;
END_VAR

IF divisor = 0 THEN
 CheckDivReal:=1;
ELSE
 CheckDivReal:=divisor;
END_IF;
```

Le résultat de la fonction CheckDivReal est utilisé comme diviseur par l'opérateur DIV. Ceci permet d'éviter une division par zéro dans le programme exemplaire ci-après car la valeur du diviseur (d) est changée de 0 à 1. Le résultat "erg" de la division est alors 799.

```
PROGRAM PLC_PRG
VAR
 erg:REAL;
 v1:REAL:=799;
```

```
 d:REAL;
END_VAR
erg:= v1 DIV d
```

MOD

Division modulo d'une variable de type BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, REAL et LREAL par une autre variable d'un ce ces types. Cette fonction donne comme résultat le reste entier de la division.

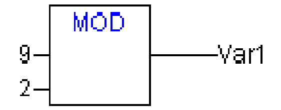
Exemple en langage IL:

```
LD 9
MOD 2
ST  var1  (* Le résultat est 1 *)
```

Exemple en langage ST:

```
var1 := 9 MOD 2;
```

Exemple en langage FBD:

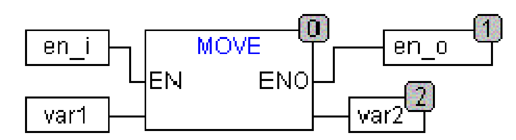


MOVE

Affectation d’une variable à une autre variable d’un type compatible. Du fait que MOVE soit disponible sous la forme d’un module dans les éditeurs FBD, CFC et LD, la fonctionnalité EN/EN0 peut également y être utilisée dans le cadre de l'affectation d’une variable.

Exemple en langage CFC en rapport avec la fonction EN/EN0 :

C’est uniquement lorsque en_i a la valeur TRUE que la valeur de la variable var1 est affectée à la variable var2.



Beispiel en langage IL:

```
LD ivar1
MOVE ivar2
ST ivar2 (* Ergebnis: var2 erhält Wert von var1 *)
```

(entspricht:

```
LD ivar1
ST ivar2 )
```

Beispiel en langage ST:

```
ivar2 := MOVE(ivar1);
```

(Ceci correspond à :

```
ivar2 := ivar1; )
```

INDEXOF

Cette fonction n’est pas prescrite par la norme CEI61131-3.
Cette fonction INDEXOF fournit comme résultat l’index interne d'un module.

Exemple en langage ST:

```
var1 := INDEXOF(module2);
```

SIZEOF

Cette fonction n'est pas prescrite par la norme CEI61131-3.

Cette fonction SIZEOF fournit comme résultat le nombre d'octets que nécessite le type de données spécifié.

Exemple en langage IL:

```
arr1:ARRAY[0..4] OF INT;
Var1 INT
LD arr1
SIZEOF
ST Var1 (* Ergebnis ist 10 *)
```

Exemple en langage ST:

```
var1 := SIZEOF(arr1);
```

9.1.2 Opérateurs sur cordons de bits

AND

Au niveau du bit AND d'opérandes binaires. Les opérandes doivent être du type BOOL, BYTE, WORD ou DWORD.

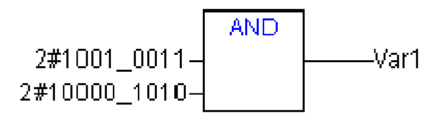
Exemple en langage IL:

```
var1 :BYTE;
LD 2#1001_0011
AND 2#1000_1010
ST  var1 (* Le résultat est 2#1000_0010 *)
```

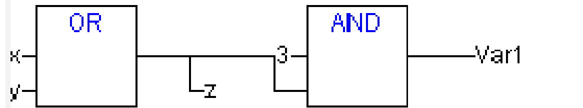
Exemple en langage ST:

```
var1 := 2#1001_0011 AND 2#1000_1010
```

Exemple en langage FBD:



Remarque : Dans le cas où vous utilisez dans FBD des générateurs de code 68xxx ou C et si vous entrez un déroulement de programme comme le suivant,



vous devez tenir compte de ce qui suit: l'affectation de la valeur de la deuxième variable d'entrée du module opérateur AND à la variable "z" ne sera pas effectuée à cause de la procédure de traitement optimalisée dans FBD dès que la variable d'entrée "a" a la valeur FALSE!

OR

Au niveau du bit OR d'opérandes binaires. Les opérandes doivent être du type BOOL, BYTE, WORD ou DWORD.

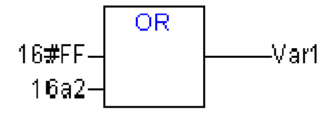
Exemple en langage IL:

```
var1 :BYTE;
LD 2#1001_0011
OR 2#1000_1010
ST  var1 (* Le résultat est 2#1001_1011 *)
```

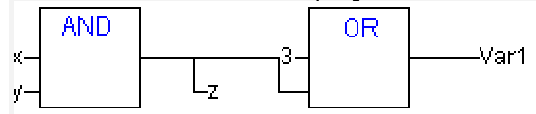
Exemple en langage ST:

```
Var1 := 2#1001_0011 OR 2#1000_1010
```

Exemple en langage FBD:



Remarque : Dans le cas où vous utilisez dans FBD des générateurs de code 68xxx ou C et si vous entrez un déroulement de programme comme le suivant,



vous devez tenir compte de ce qui suit: l'affectation de la valeur de la deuxième variable d'entrée du module opérateur OR à la variable "z" ne sera pas effectuée à cause de la procédure de traitement optimalisée dans FBD dès que la variable d'entrée "a" a la valeur TRUE!

XOR

Au niveau du bit XOR d'opérandes binaires. Les opérandes doivent être du type BOOL, BYTE, WORD ou DWORD.

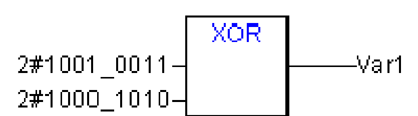
Exemple en langage IL:

```
Var1 :BYTE;
LD 2#1001_0011
XOR 2#1000_1010
ST  var1 (* Le résultat est 2#0001_1001 *)
```

Exemple en langage ST:

```
Var1 := 2#1001_0011 XOR 2#1000_1010
```

Exemple en langage FBD:



Remarque : Veuillez noter le comportement du module XOR dans sa forme étendue, avec plus de deux entrées : les entrées sont testées deux par deux et les résultats obtenus sont à nouveau comparés les uns aux autres (correspond à la norme, mais pas nécessairement aux attentes).

NOT

Au niveau du bit NOT d'un opérande binaire. L'opérande doit être du type BOOL, BYTE, WORD ou DWORD.

Exemple en langage IL:

```
Var1 :BYTE;
LD 2#1001_0011
NOT
ST  var1 (* Le résultat est 2#0110_1100 *)
```

Exemple en langage ST:

```
Var1 := NOT 2#1001_0011
```

Exemple en langage FBD:

NOT

2#1001_0011 — Var1

9.1.3 Opérateurs de décalage binaire

Remarque pour toutes les versions, y compris le Service Pack 5 de la Version 2.2: Le générateur de code pour les systèmes cible Infineon C16x exécute des opérations de calcul de décalage binaire avec Modulo 16.

SHL

Décalage vers la gauche au niveau du bit d'un opérande: erg:= SHL (in, n)

in est décalé de n bits vers la gauche, et complété avec des zéros à partir de la droite.

Les variables d'entrées erg, in et n doivent être du type BYTE, WORD ou DWORD. In est décalé de n bits vers la gauche, et complété avec des zéros à partir de la droite.

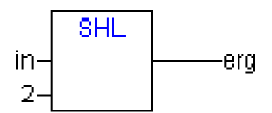
Remarque : Ne perdez pas de vue que le nombre de bits utilisé par cette opération de calcul est défini par le type de donnée de la variable d'entrée in. S'il s'agit d'une constante, le plus petit type de donnée possible est utilisé pour cette opération. Le type de donnée de la variable de sortie n'a pas d'influence sur cette opération de calcul.

Vous pouvez voir à l'aide de la représentation hexadécimale dans l'exemple suivant l'influence de l'opération sur les résultats erg_byte et erg_word pour une même valeur des variables d'entrée in_byte et in_word mais pour des types de donnée différents de la variable "in...", c.-à-d. une fois du type BYTE et l'autre fois du type WORD.

Exemple en langage ST:

```
PROGRAM shl_st
VAR
in_byte : BYTE:=16#45;
in_word : WORD:=16#45;
erg_byte : BYTE;
erg_word : WORD;
n: BYTE :=2;
END_VAR
erg_byte:=SHL(in_byte,n); (* Resultat: 16#14 *)
erg_word:=SHL(in_word;n); (* Resultat: 16#0114 *)
```

Exemple en langage FBD:



Exemple en langage IL:

```
LD   16#45
SHL  2
ST   erg_byte
```

SHR

Décalage vers la droite au niveau du bit d'un opérande: erg:= SHR (in, n)

in est décalé de n bits vers la droite. Lorsqu'un type de donnée sans signe est utilisé (BYTE, WORD, DWORD), in est complété avec des zéro à partir de la gauche. Avec un type de donnée avec signe, p.ex. INT, on procède à un décalage arithmétique, c.-à-d. qu'in est complété avec la valeur du bit supérieur.

Remarque : Ne perdez pas de vue que le nombre de bits utilisé par cette opération de calcul est défini par le type de donnée de la variable d'entrée in. S'il s'agit d'une constante, le plus petit type de donnée possible est utilisé pour cette opération. Le type de donnée de la variable de sortie n'a pas d'influence sur cette opération de calcul.

Vous pouvez voir à l'aide de la représentation hexadécimale dans l'exemple suivant l'influence de l'opération sur les résultats erg_byte et erg_word pour une même valeur des variables d'entrée

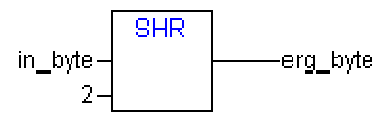
in_byte et in_word mais pour des types de donnée différents de la variable "in...", c.-à-d. une fois du type BYTE et l'autre fois du type WORD.

Exemple en langage ST:

```
PROGRAM shr_st
VAR
in_byte : BYTE:=16#45;
in_word : WORD:=16#45;
erg_byte : BYTE;
erg_word : WORD;
n: BYTE :=2;
END_VAR


erg_byte:=SHR(in_byte,n); (* Resultat 11 *)
erg_word:=SHR(in_word;n); (* Resultat 0011 *)
```

Exemple en langage FBD:



Exemple en langage IL:

```
LD  16#45
SHR 2
ST  erg_byte
```

ROL

Rotation vers la gauche au niveau du bit d'un opérande: erg:= ROL (in, n)

erg, in et n doivent être du type BYTE, WORD ou DWORD. in est déplacé n fois vers la gauche d'une position binaire tandis que le bit le plus à gauche est réinséré par la droite.

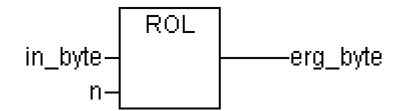
Remarque : Ne perdez pas de vue que le nombre de bits utilisé par cette opération de calcul est défini par le type de donnée de la variable d'entrée in. S'il s'agit d'une constante, le plus petit type de donnée possible est utilisé pour cette opération. Le type de donnée de la variable de sortie n'a pas d'influence sur cette opération de calcul.

Vous pouvez voir à l'aide de la représentation hexadécimale dans l'exemple suivant l'influence de l'opération sur les résultats erg_byte et erg_word pour une même valeur des variables d'entrée in_byte et in_word mais pour des types de donnée différents de la variable "in...", c.-à-d. une fois du type BYTE et l'autre fois du type WORD.

Exemple en langage ST:

```
PROGRAM rol_st
VAR
in_byte : BYTE:=16#45;
in_word : WORD:=16#45;
erg_byte : BYTE;
erg_word : WORD;
n: BYTE :=2;
END_VAR
```

Exemple en langage FBD:



Exemple en langage IL:

```
LD  16#45
ROL 2
ST  erg_byte
```

ROR

Rotation vers la droite au niveau du bit d'un opérande: erg:= ROR (IN, N)

erg, in et n doivent être du type BYTE, WORD ou DWORD. in est déplacé n fois vers la droite d'une position binaire tandis que le bit le plus à droite est réinséré par la gauche.

Remarque : Ne perdez pas de vue que le nombre de bits utilisé par cette opération de calcul est défini par le type de donnée de la variable d'entrée in. S'il s'agit d'une constante, le plus petit type de donnée possible est utilisé pour cette opération. Le type de donnée de la variable de sortie n'a pas d'influence sur cette opération de calcul.

Vous pouvez voir à l'aide de la représentation hexadécimale dans l'exemple suivant l'influence de l'opération sur les résultats erg_byte et erg_word pour une même valeur des variables d'entrée in_byte et in_word mais pour des types de donnée différents de la variable "in...", c.-à-d. une fois du type BYTE et l'autre fois du type WORD.

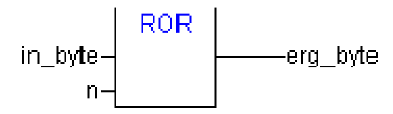
Exemple en langage ST:

```
PROGRAM ror_st

VAR
in_byte : BYTE:=16#45;
in_word : WORD:=16#45;
erg_byte : BYTE;
erg_word : WORD;
n: BYTE :=2;

END_VAR
```

Exemple en langage FBD:



Exemple en langage IL:

```
LD 16#45

ROR 2

ST erg_byte
```

9.1.4 Opérateurs de sélection

Tous les opérateurs de sélection peuvent également être appliqués à des variables. Pour ne pas alourdir la présentation, nous nous limiterons à choisir des constantes comme opérateurs dans les exemples illustrés ici.

SEL

Sélection binaire.

```
OUT := SEL(G, IN0, IN1) signifie:

OUT := IN0 if G=FALSE;

OUT := IN1 if G=TRUE.
```

IN0, IN1 et OUT peuvent être de n'importe quel type, G doit être de type BOOL. Si G a la valeur FALSE, alors le résultat de la sélection est IN0, et si G a la valeur TRUE, alors le résultat est IN1.

Exemple en langage IL:

```
LD TRUE

SEL 3,4

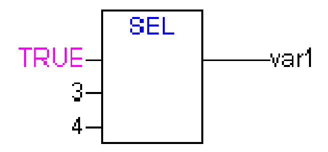
ST  Var1 (* Le résultat est 4 *)

LD FALSE

SEL 3,4

ST  Var1 (* Le résultat est 3 *)
```

Exemple en langage FBD:



Remarque: Pour optimizer le durée: Une expression avant IN0 est executée seulement si G=FALSE. Une expression avant IN1 est executée seulement si G=TRUE!
Dans la mode Simulation toutes les branches sont executées.

MAX

Fonction maximum. Fournit la plus grande des deux valeurs.

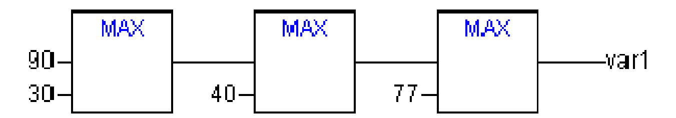
OUT := MAX(IN0, IN1)

IN0, IN1 et OUT peuvent être de n'importe quel type.

Exemple en langage IL:

```
LD 90
MAX 30
MAX 40
MAX 77
ST  Var1 (* Le résultat est 90 *)
```

Exemple en langage FBD:



MIN

Fonction minimum. Fournit la plus petite des deux valeurs.

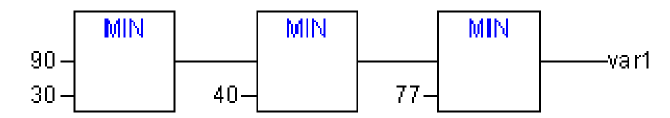
OUT := MIN(IN0, IN1)

IN0, IN1 et OUT peuvent être de n'importe quel type.

Exemple en langage IL:

```
LD 90
MIN 30
MIN 40
MIN 77
ST  Var1 (* Le résultat est 30 *)
```

Exemple en langage FBD:



LIMIT

Limitation

```
OUT := LIMIT(Min, IN, Max) signifie:
OUT := MIN (MAX (IN, Min), Max)
```

Max représente la barrière supérieure et Min la barrière inférieure du résultat. Si la valeur IN dépasse la barrière supérieure Max, alors LIMIT fournit comme valeur Max. Si IN dépasse vers le bas la barrière Min, alors le résultat est Min.

IN et OUT peuvent être de n'importe quel type.

Exemple en langage IL:

```
LD 90
LIMIT  30,80
ST  Var1 (* Le résultat est 80 *)
```

MUX

Multiplexeur

```
OUT := MUX(K, IN0,...,INn) signifie:
OUT := INK.
```

IN0, ..., INn et OUT peuvent être de n'importe quel type. K doit être du type BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT ou UDINT. MUX sélectionne dans un ensemble la K-ième valeur.

Exemple en langage IL:

```
LD 0
MUX  30,40,50,60,70,80
ST   Var1 (* Le résultat est 30 *)
```

Exemple en langage ST:

```
Var1:=MUX(0,30,40,50,60,70,80); (* Le résultat est 30 *);
```

Remarque : Afin d'optimiser l'exécution, seule l'expression qui est avant INK est calculée ! Dans une simulation par contre, toutes les séquences sont calculées.

9.1.5 Opérateurs de comparaison

GT

Plus grand que.

Un opérateur booléen qui fournit comme résultat TRUE, lorsque le premier opérande est plus grand que le deuxième. Les opérandes peuvent être de type BOOL, BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, REAL, LREAL, TIME, DATE, TIME_OF_DAY, DATE_AND_TIME ou STRING.

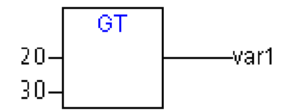
Exemple en langage IL:

```
LD 20
GT 30
ST  Var1 (* Le résultat est FALSE *)
```

Exemple en langage ST:

```
VAR1 := 20 > 30 > 40 > 50 > 60 > 70;
```

Exemple en langage FBD:



LE

Plus petit ou égal.

Un opérateur booléen qui fournit comme résultat TRUE, lorsque le premier opérande est plus petit que le deuxième ou égal au deuxième. Les opérandes peuvent être de type BOOL, BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, REAL, LREAL, TIME, DATE, TIME_OF_DAY, DATE_AND_TIME ou STRING.

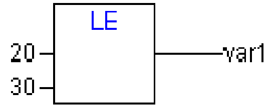
Exemple en langage IL:

```
LD 20
LE 30
ST  Var1 (* Le résultat est TRUE *)
```

Exemple en langage ST:

```
VAR1 := 20 <= 30;
```

Exemple en langage FBD:



GE

Plus grand ou égal.

Un opérateur booléen qui fournit comme résultat TRUE, lorsque le premier opérande est plus grand que le deuxième ou égal au deuxième. Les opérandes peuvent être de type BOOL, BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, REAL, LREAL, TIME, DATE, TIME_OF_DAY, DATE_AND_TIME ou STRING.

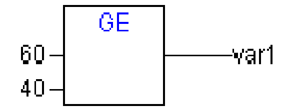
Exemple en langage IL:

```
LD 60
GE 40
ST  Var1 (* Le résultat est TRUE *)
```

Exemple en langage ST:

```
VAR1 := 60 >= 40;
```

Exemple en langage FBD:



EQ

Egalité.

Un opérateur booléen qui fournit comme résultat TRUE, lorsque les deux opérandes sont égaux. Les opérandes peuvent être de type BOOL, BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, REAL, LREAL, TIME, DATE, TIME_OF_DAY, DATE_AND_TIME ou STRING.

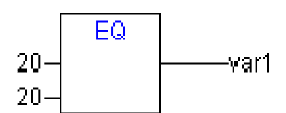
Exemple en langage IL:

```
LD 40
EQ 40
ST  Var1 (* Le résultat est TRUE *)
```

Exemple en langage ST:

```
VAR1 := 40 = 40;
```

Exemple en langage FBD:



NE

Inégalité.

Un opérateur booléen qui fournit comme résultat TRUE, lorsque les deux opérandes ne sont pas égaux. Les opérandes peuvent être de type BOOL, BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, REAL, LREAL, TIME, DATE, TIME_OF_DAY, DATE_AND_TIME ou STRING.

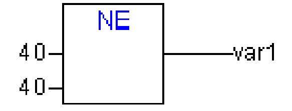
Exemple en langage IL:

```
LD 40
NE 40
ST  Var1 (* Le résultat est FALSE *)
```

Exemple en langage ST:

```
VAR1 := 40 <> 40;
```

Exemple en langage FBD:



LT

Plus petit que.

Un opérateur booléen qui fournit comme résultat TRUE, lorsque le premier opérande est plus petit que le deuxième. Les opérandes peuvent être de type BOOL, BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, REAL, LREAL, TIME, DATE, TIME_OF_DAY, DATE_AND_TIME ou STRING.

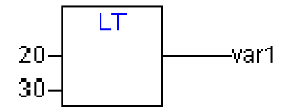
Exemple en langage IL:

```
LD 20
LT 30
ST  Var1 (* Le résultat est TRUE *)
```

Exemple en langage ST:

```
VAR1 := 20 < 30;
```

Exemple en langage FBD:



9.1.6 Opérateurs d'adressage

ADR

Cette fonction d'adressage n'est par prescrite par la norme IEC61131-3.

ADR fournit l'adresse de son argument dans un DWORD. L'adresse peut être envoyée vers des fonctions du fabricant, où elle sera traitée comme un pointeur, ou bien elle peut être affectée à un pointeur à l'intérieur du projet.

Exemple en langage ST:

```
dwVar:=ADR(bVAR);
```

Exemple en langage IL:

```
LD var1
ADR
ST  var2
man_fun1
```

Attention

BITADR

Cette fonction d'adressage n'est par prescrite par la norme IEC61131-3.

BITADR fournit le décalage du bit au sein d'un segment dans un DWORD. Ne perdez pas de vue que le décalage dépend de l'activation ou non de l'option Adressage par octets dans la configuration du système cible.

```
VAR
 var1 AT %IX2.3:BOOL;
 bitoffset: DWORD;
END_VAR
```

Exemple en langage ST:

```
bitoffset:=BITADR(var1); (* Résultat avec Adressage par octets=TRUE: 19, et avec
Adressage par octets=FALSE: 35 *)
```

Exemple en langage IL:

```
LD Var1
BITADR
ST Var2
```

Attention

Opérateur de contenu

La suppression de la référence d'un pointer se fait au moyen de l'opérateur de contenu "^" placé derrière l'identificateur du pointer.

Exemple en langage ST:

```
pt:POINTER TO INT;
var_int1:INT;
var_int2:INT;
pt := ADR(var_int1);
var_int2:=pt^;
```

Attention

9.1.7 Opérateur d'appeler

CAL

Appel d'un bloc fonctionnel.

En langage IL, CAL permet d'appeler une instance de bloc fonctionnel. Le nom de l'instance d'un bloc fonctionnel est suivi, entre parenthèses, de l'assignation des variables d'entrée du bloc fonctionnel.

Exemple :

Appel de l'instance Inst d'un bloc fonctionnel avec 0 ou TRUE comme assignations des variables Par1, Par2.

```
CAL INST(PAR1 := 0, PAR2 := TRUE)
```

9.1.8 Conversions de types

Il n'est pas permis de convertir implicitement un type «plus grand » en un type « plus petit » (par exemple la conversion de INT en BYTE ou la conversion de DINT en WORD). Si l'on veut effectuer une conversion de ce genre, il faut utiliser des fonctions particulières qui sont les conversions de type. En principe, on peut convertir n'importe quel type élémentaire en un autre type élémentaire.

Syntaxe:

```
<Type élémentaire 1>_TO_<Type élémentaire 2>
```

Conversions BOOL_TO

Conversion du type BOOL en un autre type:

Dans le cas des types numériques, le résultat est 1 si l'opérande a la valeur TRUE et 0 si l'opérande a la valeur FALSE.

Dans le cas du type STRING, le résultat est 'TRUE' ou 'FALSE', selon le cas.

voici aussi Remarques

Exemples en langage IL:

```
LD   TRUE                     (* Le résultat est 1 *)
BOOL_TO_INT
ST   I

LD TRUE                       (* Le résultat est 'TRUE' *)
BOOL_TO_STRING
ST str

LD TRUE                       (* Le résultat est T#1ms *)
BOOL_TO_TIME
ST t

LD TRUE                       (* Le résultat est TOD#00:00:00.001 *)
BOOL_TO_TOD
ST

LD FALSE                      (* Le résultat est D#1970-01-01 *)
BOOL_TO_DATE
ST dat

LD TRUE                       (* Le résultat est DT#1970-01-01-00:00:01 *)
BOOL_TO_DT
ST dandt
```

Exemples en langage ST:

```
i:=BOOL_TO_INT(TRUE);         (* Le résultat est 1 *)

str:=BOOL_TO_STRING(TRUE);    (* Le résultat est 'TRUE' *)

t:=BOOL_TO_TIME(TRUE);        (* Le résultat est T#1ms *)

tof:=BOOL_TO_TOD(TRUE);       (* Le résultat est TOD#00:00:00.001 *)

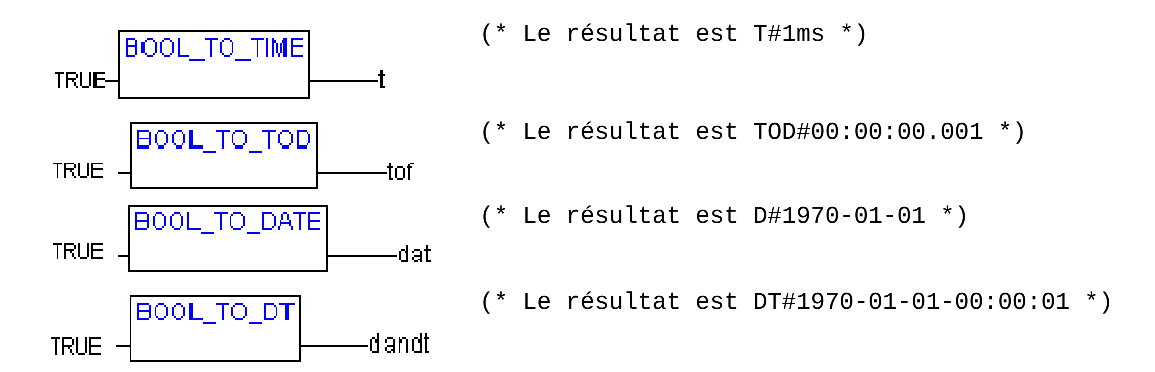
dat:=BOOL_TO_DATE(FALSE);     (* Le résultat est D#1970-01-01 *)

dandt:=BOOL_TO_DT(TRUE);      (* Le résultat est DT#1970-01-01-00:00:01 *)
```

Exemples en langage FBD:

TRUE — BOOL_TO_INT — i (* Le résultat est 1 *)

TRUE — BOOL_TO_STRING — str (* Le résultat est 'TRUE' *)



Conversions TO_BOOL

Conversion d'un type quelconque en un type BOOL:

Le résultat est TRUE si l'opérande est différent de 0. Le résultat est FALSE si l'opérande est égal à 0.

Dans le cas du type STRING, le résultat est égal à TRUE si l'opérande est 'TRUE', sinon le résultat est FALSE.

voici aussi Remarques

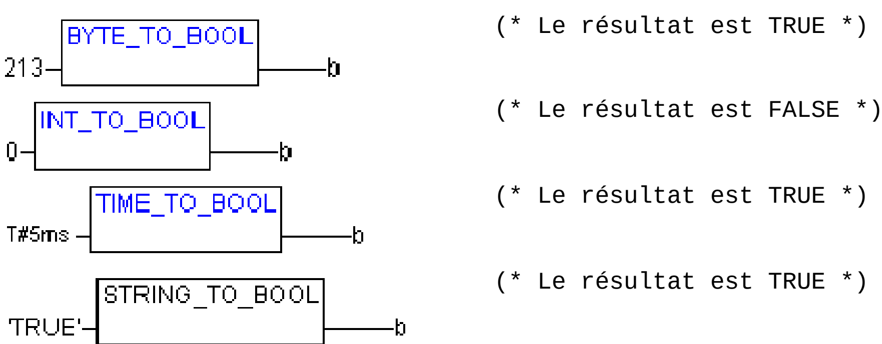
Exemples en langage IL:

```
LD 213                            (* Le résultat est TRUE *)
BYTE_TO_BOOL
ST b
LD 0                              (* Le résultat est FALSE *)
INT_TO_BOOL
ST b
LD T#5ms                          (* Le résultat est TRUE *)
TIME_TO_BOOL
ST b
LD 'TRUE'                         (* Le résultat est TRUE *)
STRING_TO_BOOL
ST b
```

Exemples en langage ST:

```
b := BYTE_TO_BOOL(2#11010101);    (* Le résultat est TRUE *)
b := INT_TO_BOOL(0);              (* Le résultat est FALSE *)
b := TIME_TO_BOOL(T#5ms);         (* Le résultat est TRUE *)
b := STRING_TO_BOOL('TRUE');      (* Le résultat est TRUE *)
```

Exemples en langage FBD:



Conversions d'un type entier en un autre type entier

Conversion d'un type entier en un autre type entier:

Lors de la conversion d'un type plus grand en un type moins grand, des informations peuvent être perdues. Si le nombre à convertir dépasse la limite de plage, alors les premiers octets de ce nombre ne sont pas pris en compte.

voici aussi Remarques

Exemple en langage ST:

```
si := INT_TO_SINT(4223); (* Le résultat est 127 *)
```

Si vous stockez le nombre entier 4223 (16#107f en notation hexadécimale) dans une variable SINT, alors celle-ci assume la valeur 127 (16#7f en notation hexadécimale).

Exemple en langage IL:

```
LD 2
INT_TO_REAL
MUL
```

Exemple en langage FBD:

```
     INT_TO_SINT
4223-           -si
```

Conversions REAL_TO/ LREAL_TO

Conversion du type REAL ou LREAL en un autre type:

La valeur est arrondie au nombre entier, vers le haut ou vers le bas, avant d'être convertie en un autre type. Ce type doit être différent des types STRING, BOOL, REAL et LREAL.

Lors de la conversion d'un type plus grand en un type moins grand, des informations peuvent être perdues.

Ne perdez pas de vue que lors de la conversion en type STRING, le nombre maximal d’emplacements de virgules est limité à 16. Si le nombre (L)REAL contient plus d’emplacements, le 16ième emplacement est arrondi et visualisé comme tel dans la chaîne de caractères. Si la chaîne de caractères a été définie trop courte pour la valeur, on réduit en conséquence à partir de la droite.

voici aussi Remarques

Exemple en langage ST:

```
i := REAL_TO_INT(1.5); (* Le résultat est 2 *)
j := REAL_TO_INT(1.4); (* Le résultat est 1 *)
```

Exemple en langage IL:

```
LD 2.7
REAL_TO_INT
GE %MW8
```

Exemple en langage FBD:

```
    LREAL_TO_INT
1.5-            -i
```

Conversions TIME_TO / TIME_OF_DAY

Conversion du type TIME ou TIME_OF_DAY en un autre type:

La date est stockée en interne dans une variable DWORD et est exprimée en millisecondes (à partir de 00:00 heures dans le cas de TIME_OF_DAY). Cette valeur est convertie.

Lors de la conversion d'un type plus grand en un type moins grand, des informations peuvent être perdues.

Dans le cas du type STRING, le résultat est la constante de temps.

voici aussi Remarques

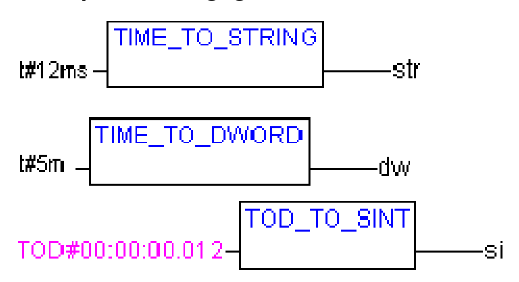
Exemples en langage IL:

```
LD T#12ms                   (* Le résultat est 'T#12ms' *)
 TIME_TO_STRING
ST str
LD T#300000ms               (* Le résultat est 300000 *)
TIME_TO_DWORD
ST dw
LD TOD#00:00:00.012         (* Le résultat est 12 *)
TOD_TO_SINT
ST si
```

Exemples en langage ST:

```
str :=TIME_TO_STRING(T#12ms);
dw:=TIME_TO_DWORD(T#5m);
si:=TOD_TO_SINT(TOD#00:00:00.012);
```

Exemples en langage FBD:



Conversions DATE_TO / DT_TO

Conversion du type DATE ou DATE_AND_TIME en un autre type:

La date est stockée en interne dans une variable DWORD et est exprimée en secondes, en comptant à partir du 1er janvier 1970. Cette valeur est convertie.

Lors de la conversion d'un type plus grand en un type moins grand, des informations peuvent être perdues.

Dans le cas du type STRING, le résultat est la constante de date.

voici aussi Remarques

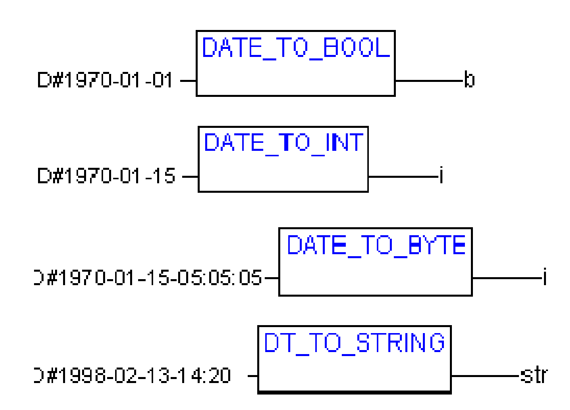
Exemples en langage IL:

```
LD D#1970-01-01             (* Le résultat est FALSE *)
DATE_TO_BOOL
ST b
LD D#1970-01-15             (* Le résultat est 29952 *)
DATE_TO_INT
ST I
LD DT#1970-01-15-05:05:05   (* Le résultat est 129 *)
DT_TO_BYTE
ST byt
LD DT#1998-02-13-14:20      (* Le résultat est
DT_TO STRING                'DT#1998-02-13-14:20' *)
ST str
```

Exemples en langage ST:

```
b :=DATE_TO_BOOL(D#1970-01-01);
i :=DATE_TO_INT(D#1970-01-15);
byt :=DT_TO_BYTE(DT#1970-01-15-05:05:05);
str:=DT_TO_STRING(DT#1998-02-13-14:20);
```

Exemples en langage FBD:



Conversions STRING_TO

Conversion du type STRING en un autre type:

L'opérande du type STRING doit avoir une valeur compatible avec le type d'arrivée, sinon le résultat est 0.

voici aussi Remarques

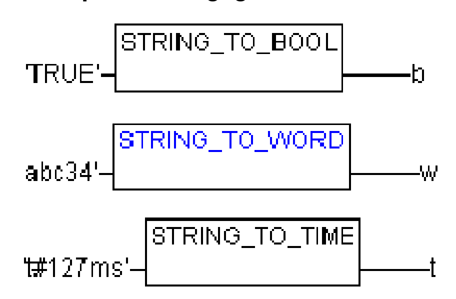
Exemples en langage IL:

```
LD 'TRUE'           (* Le résultat est TRUE *)
STRING_TO_BOOL
ST b
LD 'abc34'          (* Le résultat est 0 *)
STRING_TO_WORD
ST w
LD 't#127ms'        (* Le résultat est T#127ms *)
STRING_TO_TIME
ST t
```

Exemples en langage ST:

```
b :=STRING_TO_BOOL('TRUE');
w :=STRING_TO_WORD('abc34');
t :=STRING_TO_TIME('T#127ms');
```

Exemples en langage FBD:



TRUNC

Conversion d'un type REAL en un type INT. Seule la partie entière du nombre est retenue.

Exemple en langage IL:

```
LD 2.7
TRUNC
GE %MW8
```

Exemples en langage ST:

```
i:=TRUNC(1.9); (* Le résultat est 1 *).
i:=TRUNC(-1.4); (* Le résultat est 1 *).
```

9.1.9 Opérateurs numériques

ABS

Fournit la valeur absolue d'un nombre. ABS(-2). Les combinaisons de types suivantes sont possibles pour IN et OUT:

IN	OUT
INT	INT, REAL, WORD, DWORD, DINT
REAL	REAL
BYTE	INT, REAL, BYTE, WORD, DWORD, DINT
WORD	INT, REAL, WORD, DWORD, DINT
DWORD	REAL, DWORD, DINT
SINT	REAL
USINT	REAL
UINT	INT, REAL, WORD, DWORD, DINT, UDINT, UINT
DINT	REAL, DWORD, DINT
UDINT	REAL, DWORD, DINT, UDINT

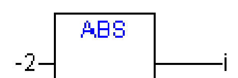
Exemple en langage IL:

```
LD   -2
ABS
ST   i   (* Ergebnis ist 2 *)
```

Exemple en langage ST:

```
i:=ABS(-2);
```

Exemple en langage FBD:



SQRT

Fournit la racine carrée d'un nombre.

IN peut être du type BYTE, WORD, DWORD, INT, DINT, REAL, SINT, USINT, UINT, UDINT; OUT doit être du type REAL.

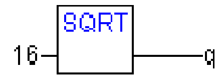
Exemple en langage IL:

```
LD   16
SQRT
ST   q (* Ergebnis ist 4 *)
```

Exemple en langage ST:

```
q:=SQRT(16);
```

Exemple en langage FBD:



LN

Fournit le logarithme naturel d'un nombre.

IN peut être du type BYTE, WORD, DWORD, INT, DINT, REAL, SINT, USINT, UINT, UDINT; OUT doit être du type REAL.

Exemple en langage IL:

```
LD  45
LN
ST  q   (* Ergebnis ist 3.80666 *)
```

Exemple en langage ST:

```
q:=LN(45);
```

Exemple en langage FBD:

45 — LN — q

LOG

Fournit le logarithme décimal d'un nombre.

IN peut être du type BYTE, WORD, DWORD, INT, DINT, REAL, SINT, USINT, UINT, UDINT; OUT doit être du type REAL.

Exemple en langage IL:

```
LD  314.5
LOG
ST  q   (* Ergebnis ist 2.49762 *)
```

Exemple en langage ST:

```
q:=LOG(314.5);
```

Exemple en langage FBD:



EXP

Fournit l'élévation exponentielle d'un nombre.

IN peut être du type BYTE, WORD, DWORD, INT, DINT, REAL, SINT, USINT, UINT, UDINT; OUT doit être du type REAL.

Exemple en langage IL:

```
LD  2
EXP
ST  q (* Ergebnis ist 7.389056099 *)
```

Exemple en langage ST:

```
q:=EXP(2);
```

Exemple en langage FBD:

2 — EXP — q

SIN

Fournit le sinus d'un nombre. Le calcul est effectué sur une valeur en radians.

IN peut être du type BYTE, WORD, DWORD, INT, DINT, REAL, SINT, USINT, UINT, UDINT; OUT doit être du type REAL.

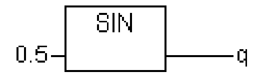
Exemple en langage IL:

```
LD      0.5
SIN
ST      q          (* Le résultat est 0,479426 *)
```

Exemple en langage ST:

```
q:=SIN(0.5);
```

Exemple en langage FBD:



COS

Fournit le cosinus d'un nombre. Le calcul est effectué sur une valeur en radians.

IN peut être du type BYTE, WORD, DWORD, INT, DINT, REAL, SINT, USINT, UINT, UDINT; OUT doit être du type REAL.

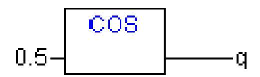
Exemple en langage IL:

```
LD  0.5
COS
ST  q  (* Ergebnis ist 0.877583 *)
```

Exemple en langage ST:

```
q:=COS(0.5);
```

Exemple en langage FBD:



TAN

Fournit la tangente d'un nombre. Le calcul est effectué sur une valeur en radians. IN peut être du type BYTE, WORD, DWORD, INT, DINT, REAL, SINT, USINT, UINT, UDINT; OUT doit être du type REAL.

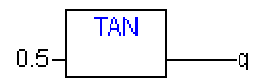
Exemple en langage IL:

```
LD  0.5
TAN
ST  q  (* Ergebnis ist 0.546302 *)
```

Exemple en langage ST:

```
q:=TAN(0.5);
```

Exemple en langage FBD:



ASIN

Fournit l'arcsinus d'un nombre (fonction inverse de la fonction sinus). La valeur est calculée en radians.

IN peut être du type BYTE, WORD, DWORD, INT, DINT, REAL, SINT, USINT, UINT, UDINT; OUT doit être du type REAL.

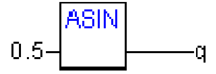
Exemple en langage IL:

```
LD  0.5
ASIN
ST  q  (* Ergebnis ist 0.523599 *)
```

Exemple en langage ST:

```
q:=ASIN(0.5);
```

Exemple en langage FBD:



ACOS

Fournit l'arccosinus d'un nombre (fonction inverse de la fonction cosinus). La valeur est calculée en radians.

IN peut être du type BYTE, WORD, DWORD, INT, DINT, REAL, SINT, USINT, UINT, UDINT; OUT doit être du type REAL.

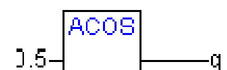
Exemple en langage IL:

```
LD 0.5
ABS
ST q  (* Ergebnis ist 1.0472 *)
```

Exemple en langage ST:

```
q:=ACOS(0.5);
```

Exemple en langage FBD:



ATAN

Fournit l'arctangente d'un nombre (fonction inverse de la fonction tangente). La valeur est calculée en radians.

IN peut être du type BYTE, WORD, DWORD, INT, DINT, REAL, SINT, USINT, UINT, UDINT; OUT doit être du type REAL.

Exemple en langage IL:

```
LD 0.5
ABS
ST q  (* Ergebnis ist 0.463648 *)
```

Exemple en langage ST:

```
q:=ATAN(0.5);
```

Exemple en langage FBD:

ATAN
0.5 — q

EXPT

Fournit l'élévation d'une variable à la puissance d'une autre variable:

OUT = IN1IN2.

IN1 et IN2 peuvent être du type BYTE, WORD, DWORD, INT, DINT, REAL, SINT, USINT, UINT, UDINT; OUT doit être du type REAL.

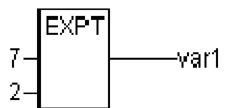
Exemple en langage IL:

```
LD 7
EXPT 2
ST var1 (* Ergebnis ist 49 *)
```

Exemple en langage ST:

```
var1 := (7,2);
```

Exemple en langage FBD:



Appendice B Les opérandes dans CoDeSys

Dans CoDeSys, vous pouvez utiliser des Constantes, des Variables, des Adresses et éventuellement des Appels de fonctions comme opérandes.

9.2 Constantes

Constantes BOOL

Les constantes booléennes (BOOL) sont les valeurs de vérité TRUE et FALSE.

Constantes TIME

CoDeSys permet la déclaration de constantes TIME. Celles-ci visent particulièrement à exploiter les temporisateurs de la bibliothèque standard. Une constante de temps TIME est toujours introduite par la lettre "t" ou "T" (ou le mot "time" ou "TIME") suivi(e) d'un dièse "#".

Ensuite vient la déclaration de temps à proprement parler; elle peut être formée de jours (désignés par "d"), d'heures (désignées par "h"), de minutes (désignées par "m"), de secondes (désignées par "s") et de millisecondes (désignées par "ms"). Il faut veiller à ce que les données de temps soient rangées par ordre décroissant (d avant h avant m avant s avant m avant ms). A noter que les périodes temporelles ne doivent pas apparaître toutes ensemble.

Exemples de constantes TIME correctes dans une affectation en langage ST:

```
TIME1 := T#14ms;
TIME1 := T#100S12ms;     (*Dépassement sur le rang supérieur
                         autorisé*)
TIME1 := t#12h34m15s;
```

Exemples erronés:

```
TIME1 := t#5m68s;        (*Dépassement sur le rang inférieur*)
TIME1 := 15ms;           (*Omission de T#*)
TIME1 := t#4ms13d;       (*Ordre incorrect des données*)
```

Constantes DATE

Ce type de constante permet de constituer des données de date. Une déclaration de constante DATE est introduite par la lettre "d" ou "D", ou le mot "DATE" ou "date" et suivi(e) d'un dièse "#". Après quoi, vous êtes libre d'entrer une date quelconque sous la forme Année-Mois-Jour.

Exemples:

```
DATE#1996-05-06
d#1972-03-29
```

Constantes TIME_OF_DAY

Ce type de constante permet d'enregistrer les heures du jour. Une déclaration de constante TIME_OF_DAY est introduite par "tod#", "TOD#", "TIME_OF_DAY#" ou "time_of_day#"; vous pouvez ensuite entrer une heure sous la forme Heure:Minute:Seconde. Les secondes peuvent être entrées sous la forme de nombres réels, autorisant donc l'entrée de fractions de seconde.

Exemples:

```
TIME_OF_DAY#15:36:30.123
tod#00:00:00
```

Constantes DATE_AND_TIME

Les constantes de date et d'heure du jour peuvent être combinées; on les appelle alors des constantes DATE_AND_TIME. Elles sont introduites par "dt#", "DT#", "DATE_AND_TIME#" ou "date_and_time#". La date est suivie d'un trait d'union puis de l'heure du jour.

Exemples:

```
DATE_AND_TIME#1996-05-06-15:36:30
dt#1972-03-29-00:00:00
```

Constantes numériques

Les libellés numériques peuvent se présenter sous la forme de chiffres binaires, de chiffres octaux, de nombres décimaux et de nombres hexadécimaux. Si un libellé entier n'est pas un nombre décimal, vous devez faire suivre sa base d'un dièse (#) avant d'écrire les nombres de la constante. Dans le cas des nombres hexadécimaux, les nombres compris entre 10 et 15 sont conventionnellement représentés par les lettres A-F.

Les caractères de soulignement uniques (_) au sein d'une valeur numérique sont autorisés.

Exemples:

```
14          (nombre décimal)
2#1001_0011 (nombre binaire)
8#67        (nombre octal)
16#A        (nombre hexadécimal)
```

Ces libellés numériques peuvent être du type BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, REAL ou LREAL.

Les conversions implicites des types "supérieurs" vers les types "inférieurs" sont interdites. Impossible donc d'utiliser une variable DINT comme une variable INT, par exemple. Pour cela, il faut faire appel aux fonctions de conversion de types, disponibles dans la bibliothèque standard .

Constantes REAL / LREAL

Les constantes REAL et LREAL peuvent être définies sous la forme de fractions décimales et de représentations exponentielles. C'est la transcription américaine qui a été choisie (la virgule est remplacée par un point).

Exemple :

```
7.4 au lieu de 7,4
1.64e+009 au lieu de 1,64e+009
```

Constantes STRING

Une chaîne est une chaîne de caractères quelconque. Les constantes STRING sont délimitées par des guillemets simples. Il est possible de saisir également des espaces et des trémas. Ces caractères ont le même statut que les autres caractères.

Dans les chaînes de caractères, la combinaison du signe dollar ($) et de deux chiffres hexadécimaux est interprétée comme la représentation hexadécimale du code de caractère à huit bits. En outre, lors de leur apparition dans une chaîne de caractères, des combinaisons de deux caractères commençant par un signe dollar sont interprétées comme suit:

$$	Signe dollar
$'	Guillemet simple
$L ou $l	Saut de ligne
$N ou $n	Nouvelle ligne
$P ou $p	Saut de page
$R ou $r	Retour automatique à la ligne
$T ou $t	Tabulation

Exemples:

```
'w1Wüß?'
'Susi et Claus'
':-)'
```

Constantes typées (Typed Literals)

En principe, le plus petit type de donnée est mis en œuvre lors de l'emploi de constantes CEI. Typed Literals vous permet d'utiliser un autre type de données sans pour autant devoir déclarer la constante de manière explicite. La constante est alors pourvue d'un préfixe qui détermine le type :

le format est comme suit : <Type>#<Literal>

<Type> indique le type de donnée, et les entrées possibles sont : BOOL, SINT, USINT, BYTE, INT, UINT, WORD, DINT, UDINT, DWORD, REAL, LREAL. Le type doit être saisi en majuscules.

<Literal> indique la constante. L'entrée doit correspondre au type de donnée indiqué par <Type>.

Exemple :

```
var1:=DINT#34;
```

Si la constante ne peut être transmise au type cible sans perte de données, un message d'erreur s'affiche :

Les constantes typées peuvent être utilisées partout où des constantes normales peuvent l'être.

9.3 Variables

Les variables peuvent être déclarées soit localement, dans la partie de déclaration d'un module, soit globalement, dans les listes de variables.

Remarque: Il est possible de définir une variable locale avec le même nom qu'une variable globale. Au sein d'un module, la variable définie localement a toujours priorité. Il n'est pas possible d'attribuer le même nom à deux variables globales ; cela provoque par exemple une erreur de compilation si une variable "var1" est définie aussi bien dans la liste des variables que dans la configuration de l'automate.

En ce qui concerne les identificateurs de variables, il faut veiller à ce qu'ils ne comportent ni espace ni tréma ni accent, à ce qu'ils ne soient pas déclarés deux fois et à ce qu'ils ne coïncident pas avec des mots-clés. Les variables peuvent s'écrire indifféremment en majuscules ou en minuscules, c.-à-d. que VAR1, Var1 et var1 représentent une seule et même variable. Les caractères de soulignement sont signifiants, c.-à-d. que "A_BCD" et "AB_CD" sont interprétés comme deux identificateurs différents. Les caractères de soulignement successifs ne sont autorisés ni au début ni à l'intérieur d'un identificateur. La longueur des identificateurs et la zone significative de variables peuvent être utilisées partout où le type déclaré le permet.

Vous pouvez appeler les variables disponibles via la Liste de sélection pour l'édition.

Drapeaux système

Les drapeaux système sont des variables déclarées implicitement, qui dépendent spécifiquement de votre automate programmable. Pour trouver les drapeaux système de votre système, choisissez l'élément de menu 'Insérer' '**Opérande**'; vous accédez à la Liste de sélection pour l'édition dans laquelle vous choisissez la catégorie **Variable système.**

Accès aux variables de tableaux, de structures et de modules

L'accès aux composants de tableaux bidimensionnels se fait comme suit:

```
<Nom_de_tableau>[Index1, Index2]
```

L'accès aux variables de structures se fait comme suit:

```
<Nom_de_structure>.<Nom_de_variable>
```

L'accès aux variables de blocs fonctionnels et de programmes se fait comme suit:

```
<Nom_de_module>.<Nom_de_variable>
```

Adressage de bits au sein de variables

On peut accéder à des bits individuels au sein de variables entières. On rattache pour ce faire l'index du bit auquel on souhaite accéder à la variable en les séparant à l'aide d'un point. L'index du bit peut être délivré par une constante au choix. L'index commence à zéro.

Exemple :

```
a : INT;
b : BOOL;
...
a.2 := b;
```

La valeur de la variable b est affectée au troisième bit de la variable a.

Si l'index est plus grand que la largeur de bit de la variable, le message d'erreur suivant s'affiche : Index '<n>' en dehors du domaine de la variable '<var>'!

L'adressage par bit est possible avec les types de variable ci-après : SINT, INT, DINT, USINT, UINT, UDINT, BYTE, WORD, DWORD.

Si le type de variable n'est pas autorisé, le message d'erreur suivant s'affiche : Type de donnée '<type>' non autorisé pour index direct.

On ne peut accéder à un bit particulier dans une variable VAR_IN_OUT.

Bitzugriff mit Hilfe einer globalen Konstante:

Si une constante globale est déclarée définissant le numéro de bit, cette constante peut être utilisée pour l'accès binaire. Les exemples suivants illustrent un tel accès binaire à une variable normale ou à une variable structurée :

Par les deux exemples: Declaration dans une liste des variables globales:

Variable enable indique, quelle bit doit être utilisee:

```
VAR_CONSTANT GLOBAL
    enable:int:=2;
END_VAR
```

Exemple 1, accès bit à une variable entière:

Déclaration dans le module :

```
VAR
    xxx:int;
END_VAR
```

Accès binaire :

`xxx.enable:=true; ->` le second bit dans la variable xxx prend la valeur TRUE

Exemple 2, accès binaire à un composant de structure entier:

Veuillez noter : L'option de projet 'Échanger constantes' (Options de compilation) doit être activée !

Déclaration dans la structure stru1:

```
TYPE stru1 :
    STRUCT
    bvar:BOOL;
    rvar:REAL;
    wvar:WORD;
{bitaccess: 'enable' 42 'Valider entraînement'}
END_STRUCT
END_TYPE
```

Déclaration dans le module :

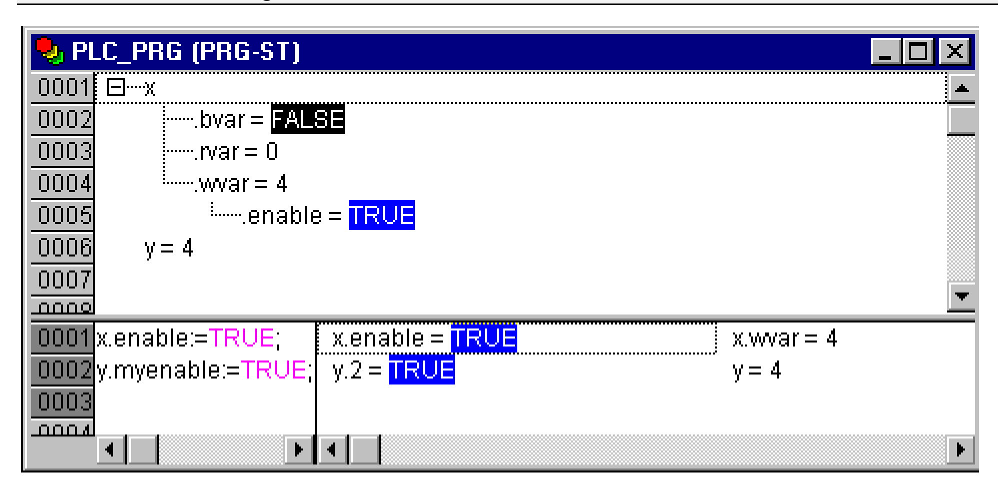
```
VAR
    x:stru1;
END_VAR
```

Accès binaire :

```
x.enable:=true;
```

Ainsi, le 42e bit dans la variable x prend la valeur TRUE. Comme bvar contient 8 bits et rvar 32, cet accès a lieu sur le 2e bit de la variable wvar, celle-ci prend alors la valeur 4.

Attention: Lors de l'espionnage, pour représenter correctement dans la liste de sélection pour l'édition et dans la fonction Intellisense, une variable qui exécute un accès binaire à une variable structurée par le biais d'une constante globale, veuillez utiliser la pragma {bitaccess}. Alors, lors de l'espionnage, la constante globale sera affichée dans la fenêtre de déclaration, en dessous de la variable structurée :



9.4 Adresses

Attention: Lorsqu'un changement En ligne est appliqué, les contenus des adresses peuvent se déplacer. Tenez-en compte lors de l'utilisation de pointeurs sur les adresses.

Adresse

La présentation directe d'éléments de mémoire individuels se fait au moyen de chaînes de caractères spécifiques, résultant de la concaténation du signe pour-cent "%", d'un préfixe de plage, d'un préfixe de dimension et d'un ou de plusieurs nombres naturels, lesquels sont séparés les uns des autres par un espace.

Les préfixes de plage suivants sont supportés:

I	Entrée
Q	Sortie
M	Mémento

Les préfixes de grandeur suivants sont supportés:

X	Bit individuel
None	Bit individuel
B	Octet (8 bits)
W	Mot (16 bits)
D	Mot double (32 bits)

Exemples:

%QX75 et %Q75	Bit de sortie 75
%IW215	Mot d'entrée 215
%QB7	Octet de sortie 7
%MD48	Mot double situé à l'emplacement de mémoire 48 de la plage des mémentos
%IW2.5.7.1	Dépend de la configuration de l'automate programmable

La validité d'une adresse dépend de la configuration d'automate actuelle du programme.

Remarque: Les valeurs booléennes sont allouées par octets, pour autant qu'elles ne renvoient pas explicitement à une adresse à bit individuel. Exemple : Une modification de la valeur de varbool1 AT %QW0 se rapporte à la zone entre QX0.0 et QX0.7.

Voir également à cet effet chapitre Appendice A, Opérateurs CEI et fonctions supplémentaires d'extension des normes, opérateurs d'adressage

Mémento

Pour accéder au mémento, il est possible d'utiliser toutes les grandeurs supportées.

Par exemple, l'adresse %MD48 concernerait les octets n°. 192, 193, 194 et 195 de la zone des mémentos (48 * 4 = 192). Le premier octet est l'octet n°0.

On peut accéder de la même façon aux mots et aux octets, voire au bits: l'adresse %MX5.0 donne accès au premier bit du cinquième mot (les bits sont en général enregistrés par groupes de mots).

Voir également à cet effet chapitre Appendice, Opérateurs CEI et fonctions supplémentaires d'extension des normes, opérateurs d'adressage

9.5 Fonctions

En langage ST, un appel de fonction peut être utilisé comme opérande.

Exemple :

```
Result := Fct(7) + 3;
```

Fonction TIME()

Cette fonction calcule le temps (milliseconds) passé depuis le start du systeme.

Type de donnée est TIME.

Example en éditeur IL:

```
TIME
ST systime  (* Ergebnis z.B.: T#35m11s342ms *)
```

Example en éditeur ST:

```
systime:=TIME();
```

Example en éditeur FBD:

```
TIME ——systime
```

Appendice C Les types de données dans CoDeSys

L'utilisateur peut utiliser des types de données standard et des types de données définis par lui-même pour la programmation. A chaque identificateur est affecté un type de données, qui détermine combien d'espace mémoire est réservé et quelles sont les valeurs contenues dans la mémoire.

9.6 Types de données standard

BOOL

Les variables de type de données **BOOL** peuvent assumer les valeurs booléennes TRUE et FALSE. Un espace mémoire de 8 bits est réservé pour ce type de variable.

Types de données entiers

Les types de données entiers sont **BYTE**, **WORD**, **DWORD**, **SINT**, **USINT**, **INT**, **UINT**, **DINT**, **UDINT**.

Les différents types de données recouvrent des plages numériques différentes. Voici les limites de plage correspondant aux types de données entiers:

Type	Limite inférieure	Limite supérieure	Espace mémoire
BYTE	0	255	8 bits
WORD	0	65535	16 bits
DWORD	0	4294967295	32 bits
SINT:	-128	127	8 bits
USINT:	0	255	8 bits
INT:	-32768	32767	16 bits
UINT:	0	65535	16 bits
DINT:	-2147483648	2147483647	32 bits
UDINT:	0	4294967295	32 bits

Pour cette raison, il se peut que des informations soient perdues lors de la conversion d'un type donné en un type plus petit.

REAL / LREAL

REAL et **LREAL** sont des "types en virgule flottante". Ils sont nécessaires lorsqu'on utilise des nombres rationnels. L'espace mémoire réservé comporte 32 bits pour REAL et 64 bits pour LREAL.

STRING

Une variable de type STRING peut recevoir une chaîne de caractère quelconque. La spécification de la dimension de l'espace mémoire réservé, au niveau de la déclaration, se rapporte aux caractères et peut se faire entre parenthèses ou entre crochets. Si aucune dimension n'est spécifiée, la valeur par défaut est de 80 caractères.

La longueur des chaînes de caractères n'est en principe pas limitée, les fonctions de chaînes de caractères n'exécutent cependant que des longueurs de 1-255 !

Exemple de déclaration de chaîne de caractères:

```
str:STRING(35):='Ceci est une chaîne';
```

Types de données de datation

Les types de données temps **TIME**, **TIME_OF_DAY** (en abrégé: **TOD**), **DATE** et **DATE_AND_TIME** (en abrégé: **DT**) sont traités en interne, de la même façon que le type de données DWORD.

Dans le cas de TIME et TOD, le temps est spécifié en millisecondes. Pour TOD, le temps est comptabilisé en partant de 00:00 heures.

Dans le cas de DATE et de DT, le temps est spécifié en secondes et est comptabilisé en partant du 1er janvier 1970 à 00:00 heures.

Vous trouverez ci-après les formats de données de datation pour l'affectation (constantes de temps et de date) :

Constantes de temps TIME

Une constante TIME est toujours introduite par la lettre "t" ou "T" (ou le mot "time" ou "TIME") suivi(e) d'un dièse "#".

Ensuite vient la déclaration de temps à proprement parler; elle peut être formée de jours (désignés par "d"), d'heures (désignées par "h"), de minutes (désignées par "m"), de secondes (désignées par "s") et de millisecondes (désignées par "ms"). Il faut veiller à ce que les données de temps soient rangées par ordre décroissant (d avant h avant m avant s avant m avant ms). A noter que les périodes temporelles ne doivent pas apparaître toutes ensemble.

Exemples de constantes TIME correctes dans une affectation en langage ST:

TIME1 := T#14ms;

TIME1 := T#100S12ms; (*Dépassement sur le rang supérieur autorisé*)

TIME1 := t#12h34m15s;

Exemples erronés:

TIME1 := t#5m68s; (*Dépassement sur le rang inférieur*)

TIME1 := 15ms; (*Omission de T#*)

TIME1 := t#4ms13d; (*Ordre incorrect des données*)

Constantes DATE, pour la date :

Une déclaration de constante DATE est introduite par la lettre "d" ou "D", ou le mot "DATE" ou "date" et suivi(e) d'un dièse "#". Après quoi, vous êtes libre d'entrer une date quelconque sous la forme Année-Mois-Jour.

Exemples :

```
DATE#1996-05-06
d#1972-03-29
```

Constantes TIME_OF_DAY, pour l'enregistrement de l'heure:

Une déclaration de constante TIME_OF_DAY est introduite par "tod#", "TOD#", "TIME_OF_DAY#" ou "time_of_day#"; vous pouvez ensuite entrer une heure sous la forme Heure:Minute:Seconde. Les secondes peuvent être entrées sous la forme de nombres réels, autorisant donc l'entrée de fractions de seconde.

Exemples :

```
TIME_OF_DAY#15:36:30.123
tod#00:00:00
```

Constantes DATE_AND_TIME, combinaison de la date et de l'heure:

Les constantes DATE_AND_TIME sont introduites par "dt#", "DT#", "DATE_AND_TIME#" ou "date_and_time#". La date est suivie d'un trait d'union puis de l'heure du jour.

Exemples :

```
DATE_AND_TIME#1996-05-06-15:36:30
dt#1972-03-29-00:00:00
```

9.7 Types de données définis

Tableau

CoDeSys supporte des tableaux (arrays) de types de données élémentaires, à une, deux ou trois dimensions. Les tableaux peuvent être définis dans la partie déclaration d'un module et dans les listes de variables globales.

Syntaxe:

<Nom_tableau>**:ARRAY** [<li1>..<ls1>,<li2>..<ls2>,<li3>..<ls3>] **OF** <Type_élém.>.

li1, li2, li3 indiquent la limite inférieure de la plage de tableau tandis que ls1, ls2 et ls3 renseignent la limite supérieure. Les valeurs limites doivent être des nombres entiers.

Exemple :

```
arr1 : ARRAY [1..13, 1..4] OF INT;
```

Initialisation de tableaux :

Exemples d'initialisation complète d'un tableau :

```
arr1 : ARRAY [1..5] OF INT := 1,2,3,4,5;
arr2 : ARRAY [1..2,3..4] OF INT := 1,3(7);
       (* bref pour 1,7,7,7 *)
arr3 : ARRAY [1..2,2..3,3..4] OF INT := 2(0),4(4),2,3;
       (* bref pour 0,0,4,4,4,4,2,3 *)
```

Exemple d'initialisation d'un tableau de structure :

```
TYPE STRUCT1
STRUCT
p1:int;
p2:int;
p3:dword;
END_STRUCT
ARRAY[1..3] OF STRUCT1:= (p1:=1,p2:=10,p3:=4723),(p1:=2,p2:=0,p3:=299),
(p1:=14,p2:=5,p3:=112);
```

Exemple d'initialisation partielle d'un tableau :

```
arr1 : ARRAY [1..10] OF INT := 1,2;
```

Les éléments pour lesquels aucune valeur n'est définie sont initialisés avec la valeur d'initialisation par défaut du type de base. Dans l'exemple ci-dessus, les éléments anarray[6] à anarray[10] sont initialisés à 0.

Accès aux composants de tableaux :

Dans le cas d'un tableau bidimensionnel, on accède aux composants du tableau à l'aide de la syntaxe suivante :

<Nom_tableau>[Index1,Index2]

Exemple :

```
arr1[9,2]
```

Remarque : Si dans votre projet vous définissez une fonction nommée CheckBounds, vous pourrez vérifier automatiquement les débordements de plage à l'intérieur de tableaux !

Fonction Checkbounds

Si dans votre projet vous définissez une fonction nommée **CheckBounds**, vous pourrez vérifier automatiquement les débordements de plage à l'intérieur de tableaux ! Le nom de la fonction est fixé et ne peut être modifié.

Exemple pour la fonction CheckBounds :

```
VAR_INPUT
 index, lower, upper: INT;
END_VAR
IF  index < lower THEN
 CheckBounds := lower;
ELSIF  index > upper THEN
 CheckBounds := upper;
ELSE  CheckBounds := index;
END_IF
```

Le programme d'exemple ci-après, destiné à tester la fonction CheckBounds, comporte un accès en dehors des limites d'un tableau défini. La fonction CheckBounds garantit que la valeur TRUE n'est pas affectée à l'endroit A[10] mais bien à la limite de zone supérieure A[7], qui, elle, est valide. La fonction CheckBounds permet de rectifier des accès en dehors des limites d'un tableau.

Programme test pour la fonction CheckBounds

```
PROGRAM PLC_PRG
VAR
 a: ARRAY[0..7] OF BOOL;
 b: INT:=10;
END_VAR
a[b]:=TRUE;
```

Pointeur

C'est dans les pointeurs que l'on enregistre les adresses de variables ou de blocs fonctionnels pendant l'exécution d'un programme.

La syntaxe des déclarations des pointeurs est la suivante:

<Identificateur>: POINTER OF <Type de données/bloc fonctionnel>;

Un pointer peut montrer sur n'importe quel type de données ou bloc fonctionnel, même ceux qui sont définis par l'utilisateur.

L'opérateur d'adressage ADR permet d'affecter au pointeur l'adresse d'une variable ou d'un bloc fonctionnel.

La suppression de la référence d'un pointer se fait au moyen de l'opérateur de contenu "^" placé derrière l'identificateur du pointer.

Exemple :

```
pt:POINTER TO INT;
var_int1:INT := 5;
var_int2:INT;
pt := ADR(var_int1);
```

```
var_int2:= pt^; (* var_int2 a maintenant la valeur 5 *)
```

Attention: Lorsqu'un changement En ligne est appliqué, les contenus des adresses peuvent se déplacer. Tenez-en compte lors de l'utilisation de pointeurs sur les adresses.

Type énumératif

Un type énumératif est un type de données défini par l'utilisateur, constitué d'un ensemble de constantes de chaîne de caractères. Ces constantes sont appelées valeurs énumératives.

Les valeurs énumératives sont connues dans tout le projet, même si elles sont déclarées localement au niveau d'un module. Créez de préférence vos types énumératifs sous forme d'objets à l'intérieur de l'Organisateur d'objets, dans l'onglet **Types de données** . Commencez par le mot clé TYPE et terminez par END_TYPE.

Syntaxe:

TYPE <Identificateur>:(<Enum_0> ,<Enum_1>, ...,<Enum_n>);
END_TYPE

L'<identificateur> peut assumer une des valeurs énumératives et est initialisé avec la première de ces valeurs. Les valeurs sont compatibles avec des nombres entiers, c.-à-d. que l'on peut effectuer des opérations comme avec INT. On peut affecter un nombre x à l'<identificateur>. Si les valeurs énumératives ne sont pas initialisées, alors le comptage part de zéro. Pour l'initialisation, tenez compte du fait que les valeurs initiales vont croissant. La validité du nombre est contrôlée lors de l'exécution.

Exemple :

```
SIGNAL: (rouge, jaune, vert:=10); (*rouge a la valeur initiale 0, jaune a la
valeur 1, vert a la valeur 10 *)
SIGNAL:=0; (* SIGNAL a la valeur rouge *)
FOR i:= rouge TO vert DO
 i := i + 1;
END_FOR;
```

Une même valeur énumérative ne peut pas être utilisée deux fois.

Exemple :

```
SIGNAL: (rouge, jaune, vert);
COULEUR: (bleu, blanc, rouge);
Erreur: "rouge" ne peut pas être utilisée pour SIGNAL et pour COULEUR.
```

Structures

Les structures sont placées sous forme d'objets (types de données définis) dans l'Organisateur d'objets, dans l'onglet **Types de données** . Commencez par le mot clé TYPE et terminez par END_TYPE.

La syntaxe des déclarations des structures est la suivante:

```
TYPE <Nom_de_structure>:
STRUCT
 <Déclaration de variable 1>
 .
 .
 <Déclaration de variable n>
END_STRUCT
END_TYPE
```

<Nom_de_structure> est un type qui est après cette déclaration connu dans tout le projet, et qui peut être utilisé comme un type de données standard.

Les structures imbriquées ne sont pas autorisées. La seule restriction est que les variables ne peuvent pas être placées sur des adresses (la déclaration AT n'est pas autorisée!).

Exemple de définition de structure nommée "polygone

```
TYPE polygone:
STRUCT
 Start:ARRAY [1..2] OF INT;
 Point1:ARRAY [1..2] OF INT;
 Point2:ARRAY [1..2] OF INT;
 Point3:ARRAY [1..2] OF INT;
 Point4:ARRAY [1..2] OF INT;
 Fin:ARRAY [1..2] OF INT;
END_STRUCT
END_TYPE
```

Exemple d'initialisation d'une structure de type Polygone:

```
Poly_1:polygone := ( Start:=3,3, Point1 =5,2, Point2:=7,3, Point3:=8,5,
Point4:=5,7, Fin := 3,5);
```

Il n'est pas possible d'initialiser avec des variables. Voir sous 'Tableau' pour un exemple d'initialisation d'un array d'une structure.

On accède aux composants de structures au moyen de la syntaxe suivante:

```
<Nom_de_structure>.<Nom_de_composant>
```

Supposons que nous ayons une structure nommée "Semaine", qui comporte un composant nommé "Lundi". Nous pouvons accéder à ce composant de la façon suivante: `Poly_1.Start`

Références

Le type de données défini par l'utilisateur Référence permet de générer un nom de rechange pour une variable, une constante ou un bloc fonctionnel.

Créez vos références sous forme d'objets à l'intérieur de l'Organisateur d'objets, dans l'onglet **Types de données** . Commencez par le mot clé TYPE et terminez par END_TYPE.

Syntaxe:

TYPE <Identificateur>**:** <Expression d'affectation>;
END_TYPE

Exemple :

```
TYPE message:STRING[50];
END_TYPE;
```

Type domaine partiel

Un type domaine partiel est un type dont la plage de valeurs ne forme qu’un sous-ensemble au sein d’un type de base. La déclaration peut se faire au sein de l'onglet Types de données, mais une variable peut aussi également être directement déclarée par un type domaine partiel :

Syntaxe pour la déclaration dans l'onglet 'Types de données':

TYPE <Nom> : <Inttype> (<li>..<ls>) END_TYPE;

<Nom>	doit être un identificateur CEI correct,
<Inttype>	est un des types de données SINT, USINT, INT, UINT, DINT, UDINT, BYTE, WORD, DWORD (LINT, ULINT, LWORD).
	est une constante qui doit être compatible avec le type de base et qui définit la

limite inférieure du domaine du type. Cette limite inférieure fait elle aussi partie de ce domaine défini.

<ls> est une constante qui doit être compatible avec le type de base et qui définit la limite supérieure du domaine du type. Cette limite supérieure fait elle aussi partie de ce domaine défini.

Exemple :

```
TYPE
 SubInt : INT (-4095..4095);
END_TYPE
```

Déclaration directe d'une variable à l'aide d'un type domaine partiel (Veillez à l'entrée correcte d'une valeur initiale lorsque le domaine partiel ne contient pas le '0') :

```
VAR
 i1 : INT (-4095..4095);
 i2: INT (5..10):=5;
 ui : UINT (0..10000);
END_VAR
```

Si un type domaine partiel se voit attribuer une constante (dans la déclaration ou l'implémentation) qui ne tombe pas dans ce domaine défini (p.ex. i:=5000), un message d'erreur s'affiche.

Pour vérifier le respect des limites du domaine partiel lors de l'exécution, vous devez insérer les fonctions **CheckRangeSigned** et **CheckRangeUnsigned**. Ces deux fonctions vous permettent de repérer les dépassements du domaine partiel de façon adéquate (la valeur peut par exemple être réduite ou un drapeau d'erreur peut être positionné). Elles sont implicitement appelées dès qu'une variable de type domaine partiel est écrite et construite à partir d'un type avec ou sans signe.

Exemple : Dans le cas d'une variable domaine partiel avec signe (comme par exemple i ci-dessus), la fonction CheckRangeSigned est appelée, pouvant être programmée pour ramener une valeur dans les limites du domaine autorisé.

```
FUNCTION CheckRangeSigned : DINT
VAR_INPUT
 value, lower, upper: DINT;
END_VAR
IF (value < lower) THEN
 CheckRangeSigned := lower;
ELSIF(value > upper) THEN
 CheckRangeSigned := upper;
ELSE
 CheckRangeSigned := value;
END_IF
```

Pour un appel automatique, le nom de fonction CheckRangeSigned est obligatoire de même que la structure de l'interface : valeur renvoyée et trois paramètres de type DINT.

La fonction est paramétrée comme suit lors de l'appel :

value:	reçoit la valeur qui devrait être attribuée au domaine partiel
lower:	limite inférieure du domaine partiel
upper:	limite supérieure du domaine partiel
Valeur renvoyée :	la valeur qui a été effectivement attribuée au domaine partiel

Avec une affectation i := 10*y;, cela va donner implicitement :

```
i := CheckRangeSigned(10*y, -4095, 4095);
```

Si par exemple y a la valeur 1000, i n'aura que la valeur 4095 après cette affectation.

Le nom de fonction et l'interface doivent naturellement être aussi corrects pour la fonction CheckRangeUnsigned.

```
FUNCTION CheckRangeUnsigned : UDINT
VAR_INPUT
 value, lower, upper: UDINT;
END_VAR
```

Attention : Si ces deux fonctions CheckRangeSigned et CheckRangeUnsigned ne sont pas disponibles, il n'y a pas de contrôle portant sur les dépassements de plage des types domaine partiel! La variable i pourrait alors obtenir n'importe quelle valeur comprise entre –32768 et 32767!

Attention : Si une fonction CheckRangeSigned ou CheckRangeUnsigned est implémentée comme ci-dessus, il se peut qu'une boucle infinie survienne dans une **boucle FOR** lors de l'utilisation de type domaine partiel. Ceci se produit précisément quand le domaine défini pour la boucle FOR est plus grand ou égal à celui du type domaine partiel.

Exemple :

```
VAR
 ui : UINT (0..10000);
END_VAR
FOR ui:=0 TO 10000 DO
...
END_FOR
```

On ne quitte pas la boucle FOR vu que ui ne peut être plus grand que 10000.

De la même manière, il faut faire attention au contenu des fonctions Checkrange lors de l'utilisation de valeurs d'incrémentation dans la boucle FOR.

Appendice D Librairies CoDeSys

9.8 Bibliothèque Standard.lib

La bibliothèque standard.lib contient les modules de base des catégories suivantes pour la programmation dans CoDeSys :

- Fonctions de chaînes de caractères
- Blocs fonctionnels bistables
- Détection de fronts
- Compteurs
- Temporisateurs

9.8.1 Fonctions de chaînes de caractères

Veuillez noter : Les fonctions de chaînes de caractère ne sont pas "thread-safe" : Lors de l'utilisation de tâches, ces fonctions de chaînes de caractères ne peuvent être utilisées que dans une seule tâche. Si la même fonction est utilisée dans plusieurs tâches, il y a un risque d'écrasement. La longueur admissible d'une chaîne de caractères lors de l'utilisation des fonctions est 1-255.

LEN

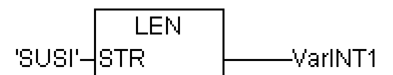
La fonction LEN (standard.lib) détermine la longueur d'une chaîne.

L'entrée STR est du type STRING et la valeur renvoyée de la fonction est du type INT.

Exemple en langage IL:

```
LD   'SUSI'
LEN
ST   VarINT1 (* Ergebnis ist 4 *)
```

Exemple en langage FBD:



Exemple en langage ST:

```
VarSTRING1 := LEN ('SUSI');
```

LEFT

La fonction LEFT (standard.lib) fournit un extrait de chaîne située à l'extrême gauche de la chaîne.

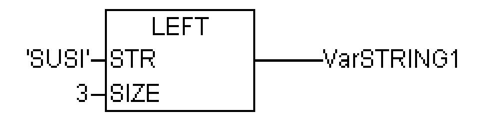
L'entrée STR est du type STRING, SIZE du type INT et la valeur renvoyée de la fonction est du type STRING.

LEFT (STR, SIZE) signifie: extraire les SIZE premiers caractères de la chaîne en partant de l'extrême gauche de la chaîne.

Exemple en langage IL:

```
LD    'SUSI'
LEFT  3
ST    VarSTRING1 (* Ergebnis ist 'SUSI' *)
```

Exemple en langage FBD:



Exemple en langage ST:

```
VarSTRING1 := LEFT ('SUSI',3);
```

RIGHT

La fonction RIGHT (standard.lib) fournit un extrait de chaîne située à l'extrême droite de la chaîne.

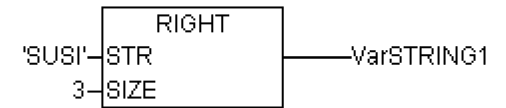
L'entrée STR est du type STRING, SIZE du type INT et la valeur renvoyée de la fonction est du type STRING.

RIGHT (STR, SIZE) signifie: extraire les SIZE premiers caractères de la chaîne en partant de l'extrême droite de la chaîne.

Exemple en langage IL:

```
LD     'SUSI'
RIGHT  3
ST     VarSTRING1 (* Ergebnis ist 'USI' *)
```

Exemple en langage FBD:



Exemple en langage ST:

```
VarSTRING1 := RIGHT ('SUSI',3);
```

MID

La fonction MID (standard.lib) fournit un extrait de chaîne située dans la chaîne.

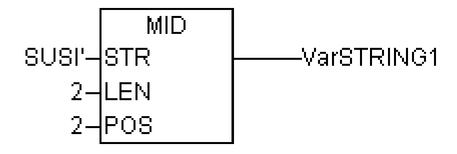
L'entrée STR est du type STRING; LEN et POS du type INT et la valeur renvoyée de la fonction est du type STRING.

MID (STR, LEN, POS) signifie: extraire LEN caractères de la chaîne STR, en commençant par le caractère positionné en POS.

Exemple en langage IL:

```
LD   'SUSI'
MID  2,2
ST   VarSTRING1 (* Ergebnis ist 'US' *)
```

Exemple en langage FBD:



Exemple en langage ST:

```
VarSTRING1 := MID ('SUSI',2,2);
```

CONCAT

La fonction CONCAT (standard.lib) fournit la concaténation (réunion) de deux chaînes.

Les entrées STR1 et STR2 et la valeur renvoyée de la fonction sont du type STRING.

Exemple en langage IL:

```
LD      'SUSI'
```

```
CONCAT 'WILLI'
ST     VarSTRING1 (* Ergebnis ist 'SUSIWILLI' *)
```

Exemple en langage FBD:

```
          CONCAT
'SUSI'-STR1          -VarSTRING1
'WILLI'-STR2
```

Exemple en langage ST:

```
VarSTRING1 := CONCAT ('SUSI','WILLI');
```

INSERT

La fonction INSERT (standard.lib) insère une chaîne dans une autre chaîne, à partir d'une position donnée.

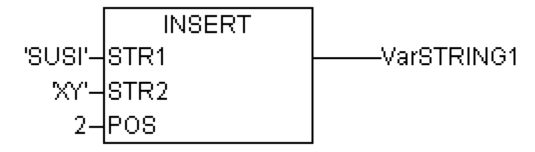
Les entrées STR1 et STR2 sont du type STRING, POS du type INT et la valeur renvoyée de la fonction est du type STRING.

INSERT(STR1, STR2, POS) signifie: insère STR2 dans STR1 à partir de la POS-ième position.

Exemple en langage IL:

```
LD     'SUSI'
INSERT 'XY',2
ST     VarSTRING1 (* Ergebnis ist 'SUXYSI' *)
```

Exemple en langage FBD:



Exemple en langage ST:

```
VarSTRING1 := INSERT ('SUSI','XY',2);
```

DELETE

La fonction DELETE (standard.lib) efface un extrait de chaîne d'une chaîne, à partir d'une position donnée.

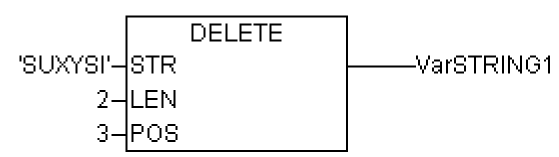
L'entrée STR est du type STRING; LEN et POS du type INT et la valeur renvoyée de la fonction est du type STRING.

DELETE(STR, LEN, POS) signifie: efface LEN caractères de la chaîne STR, en commençant par le POS-ième caractère.

Exemple en langage IL:

```
LD     'SUXYSI'
DELETE 2,3
ST     Var1 (* Ergebnis ist 'SUSI' *)
```

Exemple en langage FBD:



Exemple en langage ST:

```
Var1 := DELETE ('SUXYSI',2,3);
```

REPLACE

La fonction REPLACE (standard.lib) remplace un extrait de chaîne par un autre extrait de chaîne, à l'intérieur d'une chaîne.

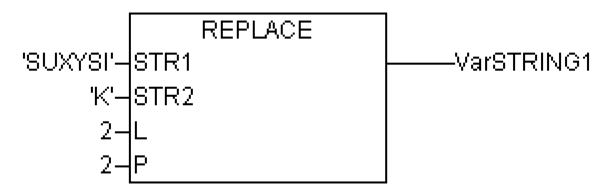
Les entrées STR1 et STR2 sont du type STRING; LEN et POS du type INT et la valeur renvoyée de la fonction est du type STRING.

REPLACE(STR1, STR2, L, P) signifie: remplace L caractères de la chaîne STR1 par la chaîne STR2, en partant du P-ième caractère.

Exemple en langage IL:

```
LD      'SUXYSI'
REPLACE 'K',2,2
ST      VarSTRING1 (* Ergebnis ist 'SKYSI' *)
```

Exemple en langage FBD:



Exemple en langage ST:

```
VarSTRING1 := REPLACE ('SUXYSI','K',2,2);
```

FIND

La fonction FIND (standard.lib) recherche un extrait de chaîne à l'intérieur d'une chaîne.

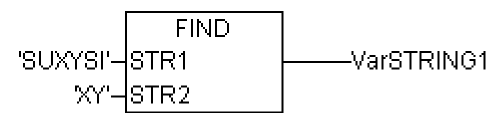
Les entrées STR1 et STR2 sont du type STRING et la valeur renvoyée de la fonction est du type INT.

FIND(STR1, STR2) signifie: recherche la position du premier caractère de la première occurrence de STR2 à l'intérieur de STR1. Si STR2 n'apparaît pas à l'intérieur de STR1, alors on a OUT := 0.

Exemple en langage IL:

```
LD   'SUXYSI'
FIND 'XY'
ST   VarINT1 (* Ergebnis ist '3' *)
```

Exemple en langage FBD:



Exemple en langage ST:

```
VarINT1 := FIND ('SUXYSI','XY');
```

9.8.2 Blocs fonctionnels bistables

SR

Bloc fonctionnel bistable (forcé dominant) (standard.lib) :

Q1 = SR (SET1, RESET) signifie:

```
Q1 = (NOT RESET AND Q1) OR SET1
```

Les entrées SET1 et RESET ainsi que la sortie Q1 sont du type BOOL.

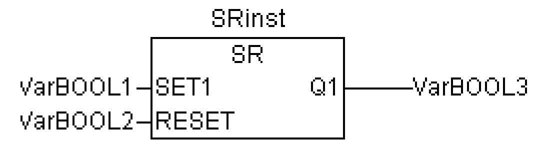
Exemple de déclaration:

```
SRInst : SR;
```

Exemple en langage IL:

```
CAL SRInst(SET1 := VarBOOL1, RESET := VarBOOL2)
LD SRInst.Q1
ST VarBOOL3
```

Exemple en langage FBD:



Exemple en langage ST:

```
SRInst(SET1:= VarBOOL1 , RESET:=VarBOOL2 );
VarBOOL3 := SRInst.Q1 ;
```

RS

Bloc fonctionnel bistable (réinitialisé dominant) (standard.lib) :

Q1 = RS (SET, RESET1) signifie:

```
Q1 = NOT RESET1 AND (Q1 OR SET)
```

Les entrées SET et RESET ainsi que la sortie Q1 sont du type BOOL.

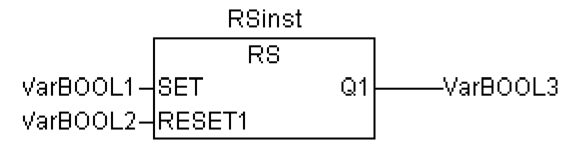
Exemple de déclaration:

```
RSInst : RS ;
```

Exemple en langage IL:

```
CAL RSInst(SET:= VarBOOL1,RESET1:=VarBOOL2)
LD RSInst.Q1
ST VarBOOL3
```

Exemple en langage FBD:



Exemple en langage ST:

```
RSInst(SET:= VarBOOL1 , RESET1:=VarBOOL2 );
VarBOOL3 := RSInst.Q1 ;
```

SEMA

Un sémaphore (interruptible) (standard.lib) .

BUSY = SEMA(CLAIM, RELEASE) signifie:

```
BUSY := X;
IF CLAIM THEN X:=TRUE;
ELSIF RELEASE THEN BUSY := FALSE; X:= FALSE;
END_IF
```

X est une variable BOOL interne, initialisée avec la valeur FALSE. Les entrées CLAIM et RELEASE ainsi que la sortie BUSY sont du type BOOL.

Lorsque SEMA est appelée et que BUSY a la valeur TRUE, cela signifie que SEMA a déjà été occupée auparavant (SEMA a été appelée avec CLAIM = TRUE). Lorsque BUSY a la valeur FALSE, cela signifie que SEMA n'a pas encore été appelée ou que SEMA a été libérée (appel avec RELEASE = TRUE).

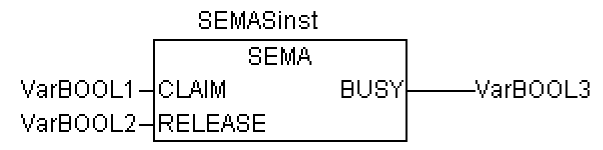
Exemple de déclaration:

```
SEMAInst : SEMA;
```

Exemple en langage IL:

```
CAL SEMAInst(CLAIM:=VarBOOL1,RELEASE:=VarBOOL2)
LD SEMAInst.BUSY
ST VarBOOL3
```

Exemple en langage FBD:



Exemple en langage ST:

```
SEMAInst(CLAIM:= VarBOOL1 , RELEASE:=VarBOOL2 );
VarBOOL3 := SEMAInst.BUSY;
```

9.8.3 Détection de fronts

R_TRIG

Le bloc fonctionnel R_TRIG (standard.lib) détecte un front montant.

```
FUNCTION_BLOCK R_TRIG
VAR_INPUT
 CLK : BOOL;
END_VAR
VAR_OUTPUT
 Q : BOOL;
END_VAR
VAR
 M : BOOL := FALSE;
END_VAR
 Q := CLK AND NOT M;
 M := CLK;
```

Tant que la variable d'entrée CLK fournit la valeur FALSE, la sortie Q et la variable auxiliaire M ont comme valeur FALSE. Dès que CLK fournit la valeur TRUE, Q fournit la valeur TRUE et ensuite M assume la valeur TRUE. Par conséquent, à l'occasion de chaque nouvel appel de l'instance du bloc fonctionnel, Q fournira à nouveau la valeur FALSE jusqu'à ce que CLK ait un front descendant suivi d'un front montant.

Exemple de déclaration:

```
RTRIGInst : R_TRIG ;
```

Exemple en langage IL:

```
CAL RTRIGInst(CLK := VarBOOL1)
LD  RTRIGInst.Q
ST  VarBOOL2
```

Exemple en langage FBD:



Exemple en langage ST:

```
RTRIGInst(CLK:= VarBOOL1);
VarBOOL2 := RTRIGInst.Q;
```

F_TRIG

Le bloc fonctionnel F_TRIG (standard.lib) détecte un front descendant.

```
FUNCTION_BLOCK F_TRIG
VAR_INPUT
 CLK: BOOL;
END_VAR
VAR_OUTPUT
 Q: BOOL;
END_VAR
VAR
 M: BOOL := FALSE;
END_VAR
 Q := NOT CLK AND NOT M;
 M := NOT CLK;
```

Tant que la variable d'entrée CLK fournit la valeur TRUE, la sortie Q et la variable auxiliaire M ont comme valeur FALSE. Dès que CLK fournit la valeur FALSE, Q fournit la valeur TRUE et ensuite M assume la valeur TRUE. Par conséquent, à l'occasion de chaque nouvel appel de l'instance du bloc fonctionnel, Q fournira à nouveau la valeur FALSE jusqu'à ce que CLK ait un front montant suivi d'un front descendant.

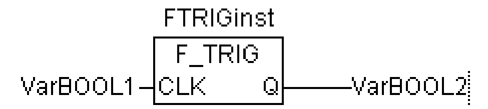
Exemple de déclaration:

```
FTRIGInst : F_TRIG ;
```

Exemple en langage IL:

```
CAL FTRIGInst(CLK := VarBOOL1)
LD  FTRIGInst.Q
ST  VarBOOL2
```

Exemple en langage FBD:



Exemple en langage ST:

```
FTRIGInst(CLK:= VarBOOL1);
VarBOOL2 := FTRIGInst.Q;
```

9.8.4 Temporisateurs

TP

Le bloc fonctionnel TP (standard.lib) est un générateur d'impulsions:

TP(IN, PT, Q, ET) signifie:

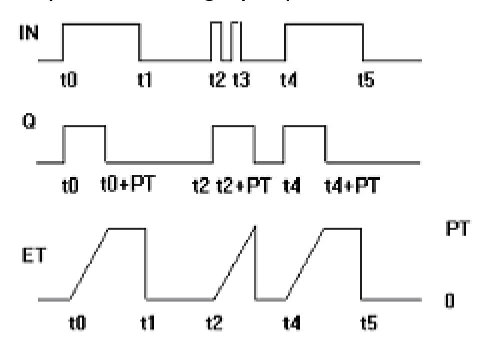
IN et PT sont des variables d'entrées du type BOOL ou TIME. Q et ET sont des variables de sortie du type BOOL ou TIME. Si IN a la valeur FALSE, alors les valeurs fournies sont FALSE ou bien 0.

Dès que IN a la valeur TRUE, la valeur de temps de ET est incrémentée en millisecondes jusqu'à ce qu'elle soit égale à la valeur de PT. Dès lors, la valeur de ET reste inchangée.

Q a la valeur TRUE si IN a la valeur TRUE et si ET est plus petit ou égal à PT. Dans tous les autres cas, Q a la valeur FALSE.

Q fournit donc un signal pour la période spécifiée dans PT.

Représentation graphique de l'évolution de TP au cours du temps:



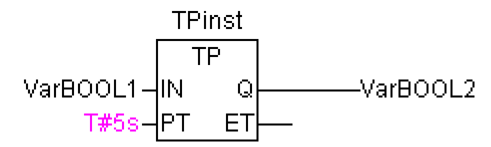
Exemple de déclaration:

```
TPInst : TP ;
```

Exemple en langage IL:

```
CAL TPInst(IN := VarBOOL1, PT := T#5s)
LD  TPInst.Q
ST  VarBOOL2
```

Exemple en langage FBD:



Exemple en langage ST:

```
TPInst(IN := VarBOOL1, PT:= T#5s);
VarBOOL2 :=TPInst.Q;
```

TON

Le bloc fonctionnel TON (Timer on-delay) (standard.lib) réalise une temporisation à l'enclenchement.

TON(IN, PT, Q, ET) signifie:

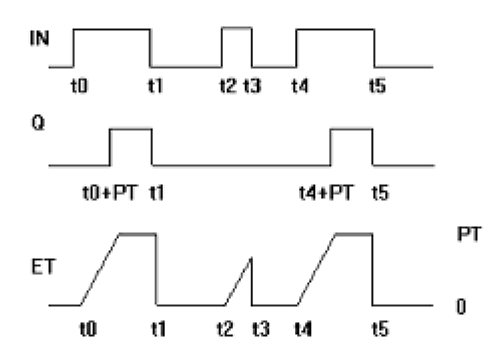
IN et PT sont des variables d'entrées du type BOOL ou TIME. Q et ET sont des variables de sortie du type BOOL ou TIME. Si IN a la valeur FALSE, alors les valeurs fournies sont FALSE ou bien 0.

Dès que IN a la valeur TRUE, la valeur de temps de ET est incrémentée en millisecondes jusqu'à ce qu'elle soit égale à la valeur de PT. Dès lors, la valeur de ET reste inchangée.

Q a la valeur TRUE si IN a la valeur TRUE et si ET est égal à PT. Dans tous les autres cas, Q a la valeur FALSE.

Q a donc un front montant lorsque le temps spécifié dans PT, qui est exprimé en millisecondes, est écoulé.

Représentation graphique de l'évolution de TON au cours du temps:



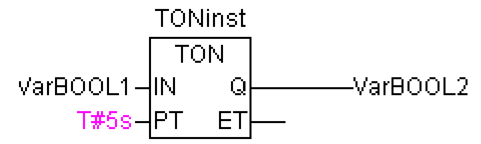
Exemple de déclaration:

```
TONInst : TON ;
```

Exemple en langage IL:

```
CAL TONInst(IN := VarBOOL1, PT := T#5s)
LD  TONInst.Q
ST  VarBOOL2
```

Exemple en langage FBD:



Exemple en langage ST:

```
TONInst(IN := VarBOOL1, PT:= T#5s);
VarBOOL2 :=TONInst.Q;
```

TOF

Le bloc fonctionnel TOF (Timer off-delay) (standard.lib) réalise une temporisation au déclenchement.

TOF(IN, PT, Q, ET) signifie:

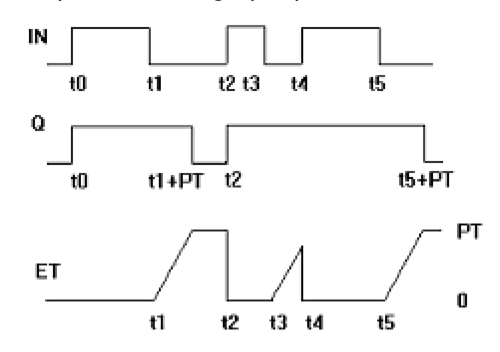
IN et PT sont des variables d'entrée du type BOOL ou TIME. Q et ET sont des variables de sortie du type BOOL ou TIME. Si IN a la valeur TRUE, alors les valeurs fournies sont TRUE ou bien 0.

Dès que IN a la valeur FALSE, la valeur de temps de ET est incrémentée en millisecondes jusqu'à ce qu'elle soit égale à la valeur de PT. Dès lors, la valeur de ET reste inchangée.

Q a la valeur FALSE si IN a la valeur FALSE et si ET est égal à PT. Dans tous les autres cas, Q a la valeur TRUE.

Q a donc un front descendant lorsque le temps spécifié dans PT, qui est exprimé en millisecondes, est écoulé.

Représentation graphique de l'évolution de TOF au cours du temps:



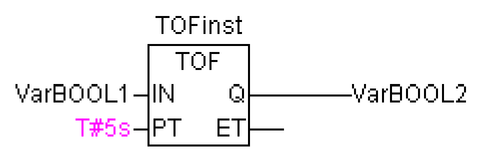
Exemple de déclaration:

```
TOFInst : TOF ;
```

Exemple en langage IL:

```
CAL TOFInst(IN := VarBOOL1, PT := T#5s)
LD TOFInst.Q
ST VarBOOL2
```

Exemple en langage FBD:

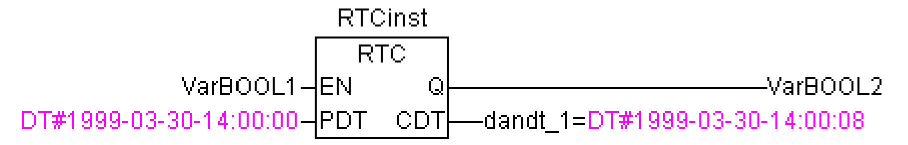


Exemple en langage ST:

```
TOFInst(IN := VarBOOL1, PT:= T#5s);
VarBOOL2 :=TOFInst.Q;
```

RTC

Le bloc fonctionnel RTC (Runtime Clock) (standard.lib) fournit continuellement la date et l'heure à partir d'un point de départ défini.



RTC(EN, PDT, Q, CDT) signifie:

EN et PDT sont des variables d'entrée du type BOOL et DATE_AND_TIME. Q et CDT sont des variables de sortie du type BOOL et DATE_AND_TIME. Si EN a la valeur FALSE, alors les sorties Q et CDT ont la valeur FALSE et DT#1970-00-00-00-00:00:00.

Dès que EN a la valeur TRUE, le temps PDT est initialisé et incrémenté en secondes et sorti à l'aide de CDT aussi longtemps que EN est TRUE (voir exemple dans le représentation ci-dessus). Dès que EN reprend la valeur FALSE, CDT reprend la valeur initiale DT#1970-01-01-00-00:00:00. Tenez compte du fait que le temps PDT n'est initialisé que par un flanc montant de EN.

9.8.5 Compteurs

CTU

Le bloc fonctionnel compteur (comptage) (standard.lib) :

Les entrées CU et RESET ainsi que la sortie Q sont du type BOOL; l'entrée PV et la sortie CV sont du type WORD.

Lorsque RESET a la valeur TRUE, la variable de comptage CV est remise à zéro. Si CU a un front montant de FALSE vers TRUE, alors CV est incrémenté d'une unité.

Q fournit la valeur TRUE si CV est supérieur ou égal à la limite supérieure de PV.

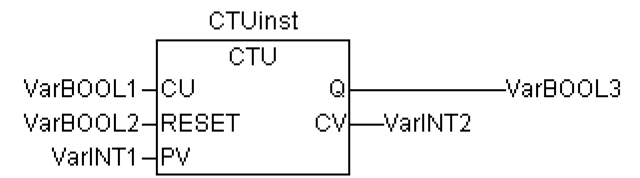
Exemple de déclaration:

```
CTUInst : CTU ;
```

Exemple en langage IL:

```
CAL   CTUInst(CU := VarBOOL1, RESET := VarBOOL2, PV := VarINT1)
LD    CTUInst.Q
ST    VarBOOL3
LD    CTUInst.CV
ST    VarINT2
```

Exemple en langage FBD:



Exemple en langage ST:

```
CTUInst(CU:= VarBOOL1, RESET:=VarBOOL2 , PV:= VarINT1);
VarBOOL3 := CTUInst.Q ;
VarINT2 := CTUInst.CV;
```

CTD

Le bloc fonctionnel compteur (décomptage) (standard.lib) :

Les entrées CD et LOAD ainsi que la sortie Q sont du type BOOL; l'entrée PV et la sortie CV sont du type WORD.

Lorsque LOAD a la valeur TRUE, la variable de comptage CV est initialisée avec la valeur de la limite supérieure PV. Si CD a un front montant de FALSE vers TRUE, alors le bloc fonctionnel CV est décrémenté d'une unité, aussi longtemps que CV est supérieur à zéro (c'est-à-dire aussi longtemps qu'il ne se produit pas de débordement par le bas).

Q fournit la valeur TRUE si CV est égal à 0.

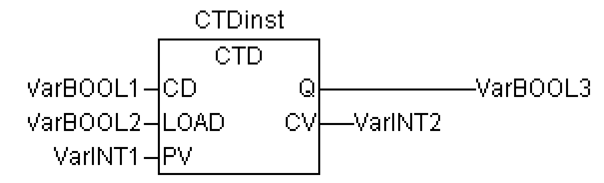
Exemple de déclaration:

```
CTDInst : CTD ;
```

Exemple en langage IL:

```
CAL   CTDInst(CD := VarBOOL1, LOAD := VarBOOL2, PV := VarINT1)
LD    CTDInst.Q
ST    VarBOOL3
LD    CTDInst.CV
ST    VarINT2
```

Exemple en langage FBD:



Exemple en langage ST:

```
CTDInst(CD:= VarBOOL1, LOAD:=VarBOOL2 , PV:= VarINT1);
VarBOOL3 := CTDInst.Q ;
VarINT2 := CTDInst.CV;
```

CTUD

Le bloc fonctionnel compteur (comptage-décomptage) (standard.lib) :

Les entrées CU, CD, RESET, LOAD et les sorties QU et QD sont du type BOOL; PV et CV sont du type WORD.

Lorsque RESET est activée, la variable de comptage CV est remise à zéro. Si LOAD est activée, alors CV est initialisée avec la valeur PV.

Si CU a un front montant de FALSE vers TRUE, alors CV est incrémenté d'une unité. Si CD a un front montant de FALSE vers TRUE, alors le bloc fonctionnel CV est diminué d'une unité, aussi longtemps que CV ne produit pas de débordement par le bas.

QU fournit la valeur TRUE si CV est supérieur ou égal à PV.

QD fournit la valeur TRUE si CV est égal à 0. (0 est la limite inférieure implémentée dans CoDeSys.)

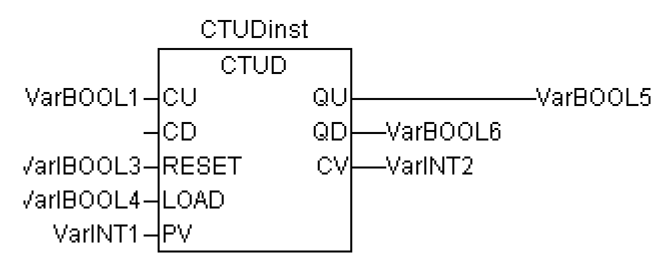
Exemple de déclaration:

```
CTUDInst : CUTD ;
```

Exemple en langage IL:

```
CAL  CTUDInst(CU:=VarBOOL2, RESET:=VarBOOL3, LOAD:=VarBOOL4, PV:=VarINT1)
LD   CTUDInst.Q
ST   VarBOOL5
LD   CTUDInst.QD
ST   VarBOOL5
LD   CTUInst.CV
ST   VarINT2
```

Exemple en langage FBD:



Exemple en langage ST:

```
CTUDInst(CU := VarBOOL1, CU:= VarBOOL2, RESET := VarBOOL3, LOAD:=VarBOOL4 , PV:=
VarINT1);
VarBOOL5 := CTUDInst.QU ;
VarBOOL6 := CTUDInst.QD ;
VarINT2 := CTUDInst.CV;
```

9.9 Bibliotheque Util.lib

Cette bibliothèque contient une série supplémentaire de modules, qui peuvent être utilisés pour la conversion BCD, les fonctions sur bits/octets et les fonctions mathématiques auxiliaires, mais aussi comme régulateurs, générateurs de signaux et manipulateurs de fonctions, ainsi que pour le traitement de valeurs analogiques.

Comme certaines de ces fonctions et modules fonctionnels contiennent des variables REAL n'étant pas supportées par certains systèmes d'exécution, il existe une bibliothèque supplémentaire UTIL_NO_REAL, qui ne reprend pas ces modules.

9.9.1 Conversion BCD

Un octet en format BCD contient des valeurs entières comprises entre 0 et 99. L'emplacement de chaque chiffre décimal occupe quatre bits, le chiffre décimal des dizaines étant mémorisé dans les bits 4-7. Par conséquent, le format BCD ressemble à la notation hexadécimale. Il s'en différencie par le fait qu'un octet BCD ne peut mémoriser que des valeurs comprises entre 0 et 99 alors qu'un octet hexadécimal va de 0 à FF.

Exemple : Le nombre entier 51 doit être converti au format BCD. En notation binaire, 5 s'écrit 0101 et 1 s'écrit 0001. Cela donne 01010001 pour l'octet BCD, ce qui correspond à la valeur $51=81.

BCD_TO_INT

Cette fonction (util.lib) permet de convertir un octet au format BCD en une valeur INT:

La valeur d'entrée de cette fonction est de type BYTE et la sortie est de type INT.

Si un octet est passé à la fonction alors qu'il n'est pas au format BCD, la valeur de sortie est –1.

Exemples en langage ST:

```
i:=BCD_TO_INT(73); (* Le résultat est 49 *)
k:=BCD_TO_INT(151); (* Le résultat est 97 *)
l:=BCD_TO_INT(15); (* Le résultat est –1 parce que pas en format BCD *)
```

INT_TO_BCD_

Cette fonction (util.lib)permet de convertir une valeur INTEGER en un octet au format BCD:

La valeur d'entrée de cette fonction est de type INT et la sortie est de type BYTE.

Si une valeur INTEGER est passée à la fonction mais qu'elle ne peut pas être convertie en un octet au format BCD, la valeur de sortie est 255.

Exemples en langage ST:

```
i:=INT_TO_BCD(49); (* Le résultat est 73 *)
k:=BCD_TO_INT(97); (* Le résultat est 151 *)
l:=BCD_TO_INT(100); (* Erreur! Sortie: 255 *)
```

9.9.2 Fonctions sur bits/octets

EXTRACT

Cette fonction (util.lib)comporte les entrées X, de type DWORD, et N, de type BYTE. La valeur de sortie est une valeur BOOL, dont le contenu correspond au n-ième bit de l'entrée X, en commençant à compter avec la bit numéro zéro.

Exemples en langage ST:

```
FLAG:=EXTRACT(X:=81, N:=4); (* Résultat: TRUE parce que 81 correspond à 1010001 en
binaire et le 4-ième bit est 1 *)
FLAG:=EXTRACT(X:=33, N:=0); (* Résultat: TRUE parce que 33 correspond à 100001 en
binaire et le 0-ième bit est 1 *)
```

PACK

Cette fonction (util.lib)peut renvoyer jusqu'à 8 bits d'entrée B0, B1, …, B7 de type BOOL sous forme d'un seul BYTE.

Le bloc fonctionnel UNPACK est étroitement lié à cette fonction.

PUTBIT

Cette fonction (util.lib) comporte les entrées X, de type DWORD, N de type BYTE et B, de type BOOL.

PUTBIT affecte le n-ième bit de X à la valeur B, en commençant à compter avec le bit numéro zéro.

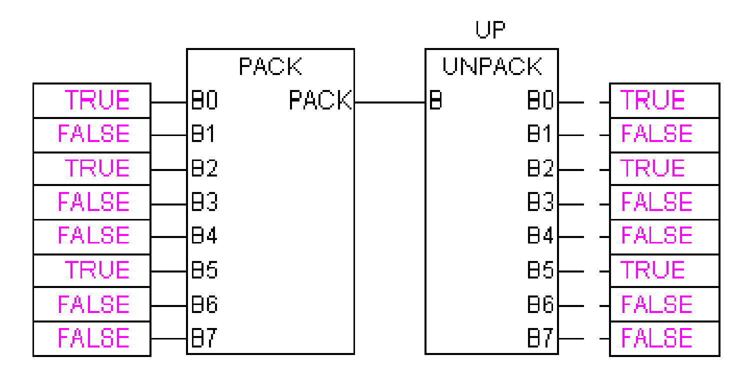
Exemple en langage ST:

```
A:=38; (* 100110 en binaire*)
B:=PUTBIT(A,4,TRUE); (* Résultat: 54 = 2#110110 *)
C:=PUTBIT(A,1,FALSE); (* Résultat: 36 = 2#100100 *)
```

UNPACK

La fonction UNPACK (util.lib) convertit l'entrée B de type BYTE en 8 variables de sortie B0, …, B7 de type BOOL. Il s'agit donc de la fonction inverse de PACK.

Exemple dans l'éditeur CFC: Sortie:



9.9.3 Fonctions mathématiques auxiliaires

DERIVATIVE

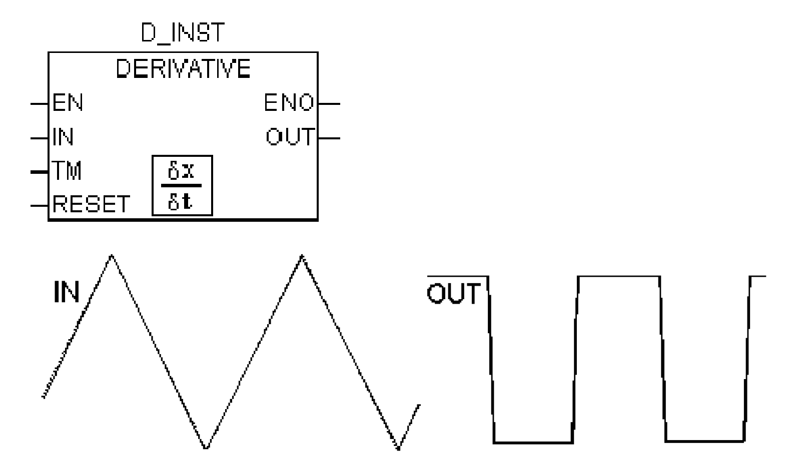
Ce bloc fonctionnel (util.lib) détermine approximativement la dérivation locale.

La valeur de la fonction est passée par l'intermédiaire de la variable IN, sous la forme REAL. TM contient le temps écoulé en msec, sous la forme DWORD tandis que l'entrée RESET, de type BOOL, permet de redémarrer le bloc fonctionnel en passant la valeur TRUE.

La sortie OUT est de type REAL.

Pour obtenir le meilleur résultat possible, DERIVATIVE effectue une approximation à partir des quatre dernières valeurs afin de restreindre autant que possible les erreurs dues à des imprécisions au niveau des paramètres d'entrée.

Module dans l'éditeur CFC:



INTEGRAL

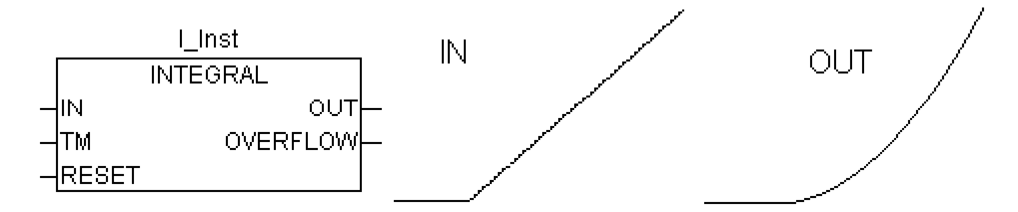
Ce bloc fonctionnel (util.lib) détermine approximativement l'intégrale d'une fonction.

De la même manière que pour DERIVATIVE, la valeur de la fonction est passée par l'intermédiaire de la variable IN, sous la forme REAL. TM contient le temps écoulé en msec, sous la forme DWORD tandis que l'entrée RESET, de type BOOL, permet de redémarrer le bloc fonctionnel en passant la valeur TRUE.

La sortie OUT est de type REAL.

Si l'intégrale atteint la limite de la plage des valeurs d'une variable REAL (env. $\pm 10^{38}$), la variable de sortie booléenne OVERFLOW est positionnée sur TRUE, et le module reste bloqué jusqu'à ce qu'il soit réinitialisé par un RESET.

Module dans l'éditeur CFC: Exemple : Intégration d'une fonction linéaire:



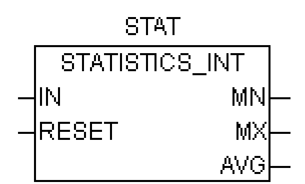
STATISTICS_INT

Ce bloc fonctionnel (util.lib) calcule certaines valeurs statistiques de référence:

L'entrée IN est de type INT. Si l'entrée RESET prend la valeur TRUE, alors toutes les valeurs sont réinitialisées.

Les sorties MN et MX contiennent respectivement la valeur minimale et la valeur maximale de IN. AVG contient la moyenne, c'est-à-dire la valeur probable de IN. Ces trois sorties sont de type INT.

Module dans l'éditeur CFC:



STATISTICS_REAL

Ce bloc fonctionnel (util.lib) correspond à STATISTICS_INT, à ceci près que l'entrée IN est de type REAL, de même que les sorties MN, MX, AVG.

VARIANCE

VARIANCE (util.lib) calcule la variance des valeurs d'entrée.

L'entrée IN est de type REAL, l'entrée RESET de type BOOL et la sortie OUT de nouveau de type REAL.

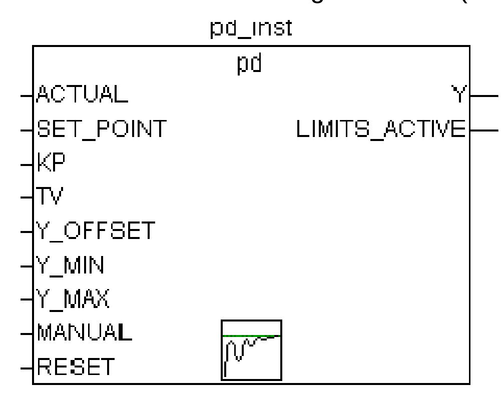
Ce bloc fonctionnel calcule la variance des valeurs d'entrée. VARIANCE est réinitialisé lorsque RESET=TRUE.

L'écart type peut être calculé facilement à partir de la racine carrée de VARIANCE.

9.9.4 Régulateurs

PD

Le bloc fonctionnel de régulateur PD (util.lib) :



ACTUAL (valeur effective) et SET_POINT (valeur de consigne), ainsi que KP, le coefficient de proportionnalité, sont des valeurs d'entrée de type REAL. TV est de type DWORD et contient la durée d'action dérivée, exprimée en msec. Y_OFFSET, Y_MIN et Y_MAX sont du type REAL et servent à la transformation de la valeur de réglage en une plage prédéfinie. MANUAL, de type BOOL, permute en mode manuel. RESET est de type BOOL et permet de réinitialiser le régulateur.

La sortie, à savoir la valeur de réglage (Y), est de type REAL, et est calculée comme suit :

```
Y = KP · (Δ + TV δΔ/δt) + Y_OFFSET dans laquelle Δ=SET_POINT-ACTUAL
```

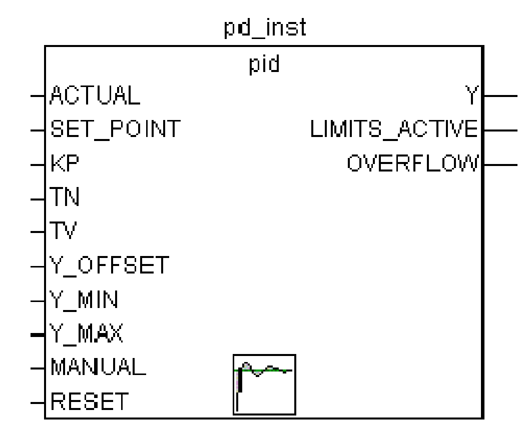
En outre, Y est restreint à la plage autorisée fixée par Y_MIN et Y_MAX. Lorsque Y dépasse ces limites, la variable de sortie booléenne LIMITS_ACTIVE obtient la valeur TRUE. Si vous ne souhaitez pas attribuer de limites à la valeur de réglage, Y_MIN et Y_MAX doivent être réglés sur 0.

Si MANUAL a la valeur TRUE, le régulateur interrompt son fonctionnement, c.-à-d. Y n’est pas changé (par le régulateur), tant que MANUAL n’obtient pas la valeur FALSE, ce qui aurait pour effet de réinitialiser le régulateur.

Un régulateur P est facilement créé, en réglant TV de manière fixe sur 0.

PID

Le bloc fonctionnel de régulateur PID (util.lib) :



La différence avec le régulateur PD réside dans le fait que ce bloc fonctionnel comporte une entrée supplémentaire DWORD TN pour la durée de réajustage en msec.

La sortie, à savoir la valeur de réglage (Y), est également de type REAL, et comporte à l'inverse du régulateur PD, une partie intégrale supplémentaire :

```
Y = KP · (Δ + 1/TN ?Δ(t)dt + TV δΔ/δt) + Y_OFFSET
```

Le régulateur PID peut facilement être converti en un régulateur PI, en entrant TV=0.

De par la partie intégrale supplémentaire, un dépassement peut survenir, en cas de paramétrage incorrect du régulateur, lorsque l'intégrale devient trop grande dû à l'erreur Δ. Par mesure de sécurité, une variable booléenne de sortie OVERFLOW est disponible, et celle-ci obtient dans ce cas la valeur TRUE. En même temps, le régulateur interrompt son fonctionnement et il est réactivé par une réinitialisation.

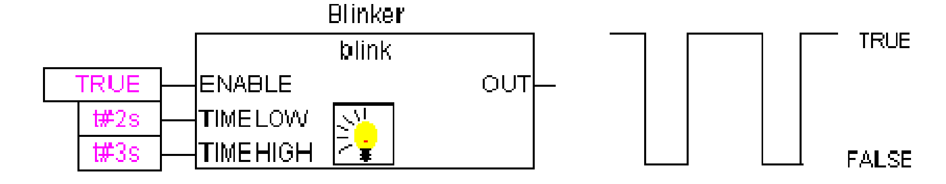
9.9.5 Générateurs de signaux

BLINK

Le bloc fonctionnel BLINK (util.lib) génère un signal pulsatoire. BLINK comporte les entrées ENABLE, de type BOOL, ainsi que TIMELOW et TIMEHIGH, de type TIME. La sortie OUT est de type BOOL.

Si la valeur TRUE est affectée à l'entrée ENABLE, alors BLINK est activé, ce qui signifie que la valeur TRUE sera affectée à la sortie pendant la durée TIMEHIGH et ensuite la valeur FALSE sera affectée à la sortie pendant la durée TIMELOW.

Exemple dans l'éditeur CFC :



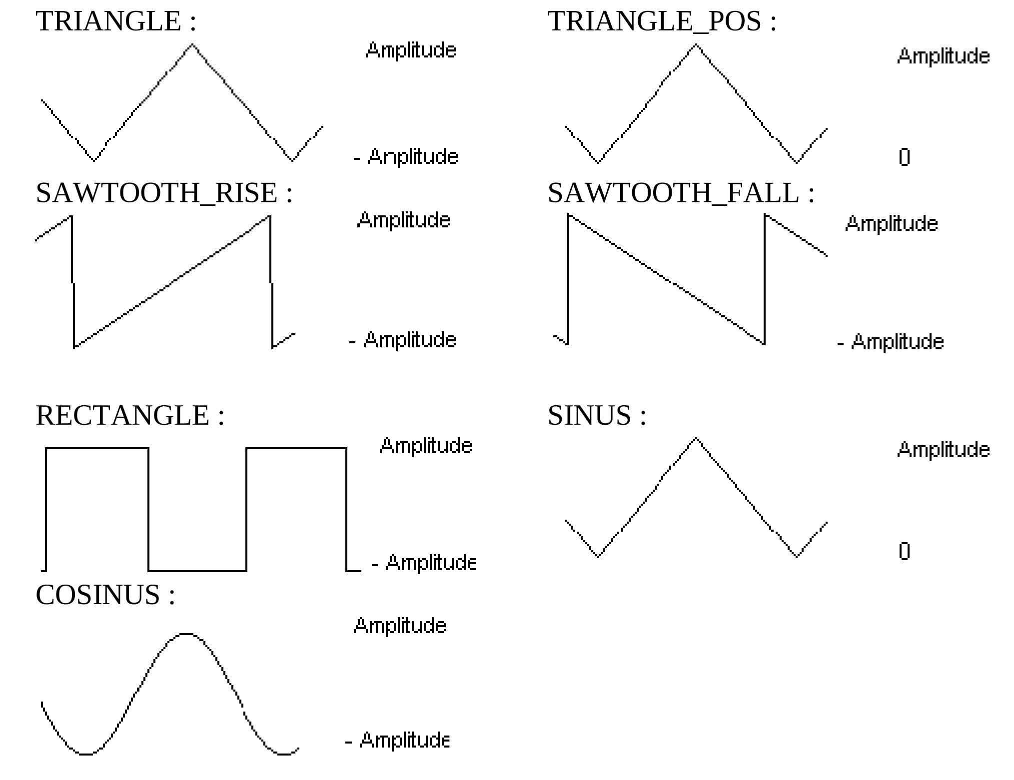
GEN

Le générateur de fonctions (util.lib) crée des fonctions périodiques typiques :

Au niveau des entrées, on trouve MODE, qui est du type d'énumération prédéfini GEN_MODE, BASE, de type BOOL, PERIOD, de type TIME, CYCLES et AMPLITUDE, deux valeurs de type INT, ainsi que RESET, une entrée de type BOOL. Il existe une sortie OUT, de type INT.

MODE définit le type de fonction à créer. Les valeurs d'énumération TRIANGLE et TRIANGLE_POS fournissent deux fonctions triangulaires, SAWTOOTH_RISE et SAWTOOTH_FALL correspondent

respectivement à une fonction en dents de scie montante et descendante, RECTANGLE donne un signal rectangulaire tandis que SINUS et COSINUS permettent d'obtenir un signal sous forme de sinus et de cosinus :



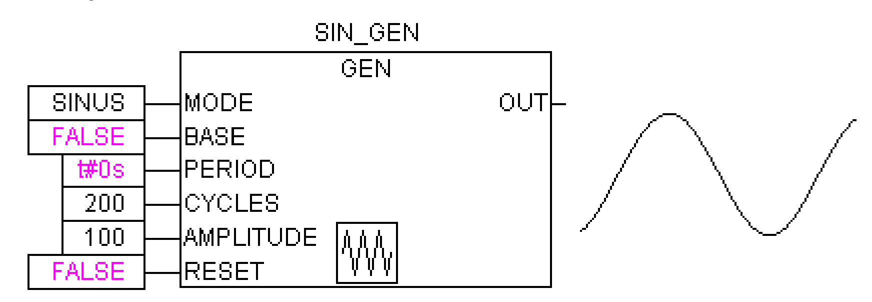
BASE indique si la durée de la période se rapporte bien à un temps prédéfini (dans ce cas, BASE=TRUE) ou plutôt à un nombre déterminé de cycles, c'est-à-dire d'appel du bloc fonctionnel (dans ce cas, BASE=FALSE).

PERIOD ou CYCLES définissent la durée de la période correspondante.

AMPLITUDE permet de définir simplement l'amplitude de la fonction à créer.

Le générateur de fonctions est réinitialisé sur 0 dès que RESET=TRUE.

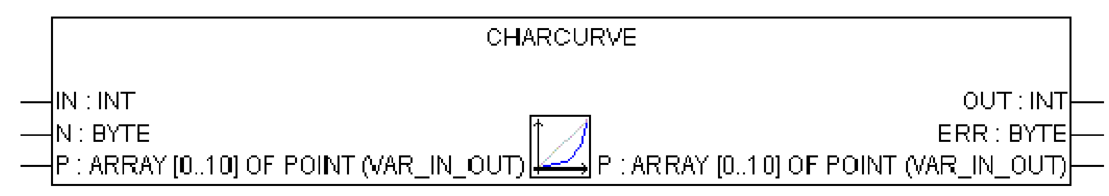
Exemple dans l'éditeur CFC :



9.9.6 Manipulateurs de fonctions

CHARCURVE

Ce bloc fonctionnel (util.lib) permet de représenter des valeurs à l'aide d'une fonction composée de segments linéaires :



IN, de type INT, est alimentée par la valeur à manipuler. N, de type BYTE, désigne le nombre de points définissant la fonction de transformation. Cette courbe caractéristique est ensuite définie dans ARRAY P[0..10], où P est de type POINT, une structure composée de deux valeurs INT (X et Y).

Au niveau de la sortie, on trouve OUT, de type INT, la valeur manipulée et ERR, de type BYTE, qui signale des erreurs éventuelles.

Les points P[0]..P[N-1] dans ARRAY doivent être triés d'après leur valeur en X, sinon la valeur 1 est affectée à ERR. Si l'entrée IN n'est pas comprise entre P[0].X et P[N-1].X, alors on obtient ERR=2 et OUT prend la valeur limite P[0].Y ou P[N-1].Y, selon le cas.

Si N se situe en dehors des valeurs admises comprises entre 2 et 11, alors on obtient ERR=4.

Exemple en langage ST :

Dans un premier temps, il faut définir ARRAY P au niveau de l'en-tête :

```
VAR
 ...
 COURBE:CHARCURVE;
 KL:ARRAY[0..10] OF POINT:=(X:=0,Y:=0),(X:=250,Y:=50),
(X:=500,Y:=150),(X:=750,Y:=400),7((X:=1000,Y:=1000));
 COMPTEUR:INT;
 ...
END_VAR
Ensuite nous alimentons CHARCURVE avec une valeur qui, par exemple, va croissant :
COMPTEUR:=COMPTEUR+10;
 COURBE(IN:=COMPTEUR,N:=5,P:=KL);
Le tracé ci-dessous permet d'illustrer cela:
```

ZÄHLER KENNLINIE OUT

RAMP_INT

RAMP_INT (util.lib) permet de limiter la montée ou la descente de la fonction alimentée.

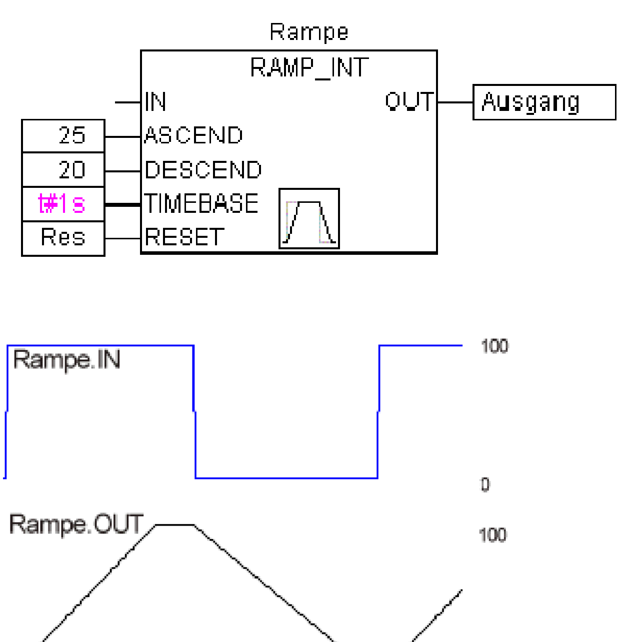
Au niveau des entrées, on trouve trois valeurs INT : IN, l'entrée pour la fonction, ainsi que ASCEND et DESCEND, qui correspondent respectivement à l'augmentation et à la diminution maximale par période. La période est définie par TIMEBASE, qui est de type TIME. Si la valeur TRUE est affectée à RESET, alors RAMP_INT est réinitialisée.

La sortie OUT, de type INT, contient la valeur de la fonction, dont la montée et la descente sont limitées.

Notez que des problèmes peuvent survenir lorsque TIMEBASE est plus petit que la durée d'un cycle complet.

Si t#0s est affecté à TIMEBASE, alors les limitations ascendante et descendante se rapportent à un cycle et donc à un appel de bloc fonctionnel.

Exemple dans l'éditeur CFC :



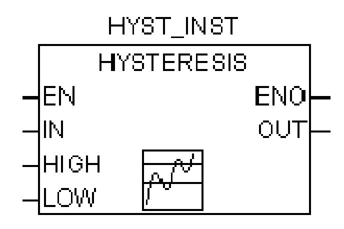
RAMP_REAL

RAMP_REAL (util.lib) fonctionne exactement comme RAMP_INT si ce n'est que IN, ASCEND, DESCEND ainsi que la sortie OUT sont de type REAL.

9.9.7 Traitement de valeurs analogiques

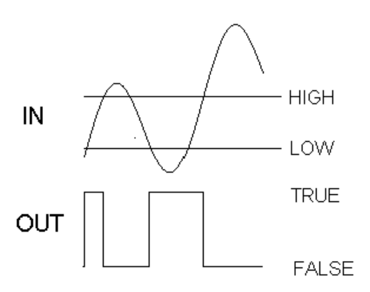
HYSTERESIS

En ce qui concerne les entrées, ce bloc fonctionnel (util.lib) comporte trois valeurs de type INT, à savoir IN, HIGH et LOW. La sortie OUT est de type BOOL.



Si IN dépasse la valeur limite LOW, alors la valeur TRUE est affectée à OUT. Si IN dépasse la limite supérieure HIGH, alors la sortie OUT fournit comme valeur FALSE.

Exemple explicatif :



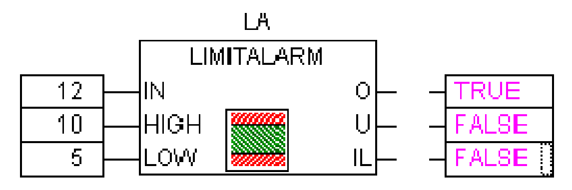
LIMITALARM

Ce bloc fonctionnel (util.lib) indique si la valeur d'entrée est comprise dans un intervalle prédéfini et fournit la limite qui fait l'objet d'un dépassement, le cas échéant.

Les valeurs d'entrée IN, HIGH et LOW sont de type INT alors que les sorties O, U et IL sont de type BOOL.

Si la limite supérieure HIGH est dépassée par IN, alors la valeur TRUE est affectée à O. Si LOW est dépassée vers le bas, alors la valeur TRUE est affectée à U. En revanche, si IN est compris entre LOW et HIGH, alors la valeur TRUE est affectée à IL.

Exemple dans l'éditeur CFC : Résultat:



9.10 Bibliotheque Analyzation.llib

Cette bibliothèque contient des modules permettant d'analyser les expressions. Lorsqu'une expression composée a la valeur globale FALSE, les composants qui ont contribué à ce résultat peuvent être déterminés. Avec éditeur SFC, le drapeau SFCErrorAnalyzationTable utilise ces fonctions de manière implicite pour analyser les expressions de transition.

Exemple d'expression :

```
b OR NOT(y < x) OR NOT (NOT d AND e)
```

Les fonctions :

Tous les modules ont les variables suivantes en commun :

InputExpr: BOOL, l'expression à analyser

DoAnalyze: BOOL, TRUE active l'analyse

ExpResult: BOOL, valeur actuelle de l'expression

Différente est la sortie du résultat de l'analyse:

AnalyzeExpression affiche dans une chaîne de caractères les composants de l'expression qui ont provoqué la valeur globale FALSE. Pour ce faire, on utilise la fonction AppendErrorString, pour que les composants individuels soient séparés par le symbole "|" dans la chaîne de caractères de sortie.

OutString: STRING, résultat de l'analyse, succession des composants concernés de l'expression (p.ex. y < x | d)

AnalyseExpressionTable écrit dans un array les composants de l'expression qui ont provoqué la valeur globale FALSE, et ExpressionResult donne les informations ci-dessous pour chaque composant de sa structure: Nom (name), adresse (address), commentaire (comment), valeur actuelle (value).

OutTable: ARRAY [0..15] OF ExpressionResult;

par exemple:

```
⊟ SFCErrorAnalyzationTable
   ⊟ SFCErrorAnalyzationTable[0]
        .name = 'b'
        .address = '%MX0.0'
        .comment = 'false'
        .value = FALSE
   ⊞ SFCErrorAnalyzationTable[1]
   ⊞ SFCErrorAnalyzationTable[2]
   ⊞ SFCErrorAnalyzationTable[3]
```

AnalyseExpressionCombined contient les fonctionnalités de AnalyzeExpression et de AnalyseExpressionTable

9.11 Bibliothèques de système

Remarque

En fonction du système cible, différentes bibliothèques de système CoDeSys sont à disposition. Reportez-vous à ce sujet à l'aperçu dans le document SysLIbs_Overview.pdf repris dans le dossier de documents de votre répertoire d'installation; vous y trouverez également des descriptions pour chaque bibliothèque.

Appendice E Aperçu: Opérateurs et modules de bibliothèques

Les tableaux ci-dessous vous donnent un aperçu des **Opérateurs et Modules de bibliothèques** qui sont disponibles dans CoDeSys ou dans les bibliothèques Standard.lib et Util.lib. Le format des éditeurs littéraux IL et ST a été ici utilisée. Pour l'éditeur littéral IL, les modificateurs disponibles ont été énumérés.

Veuillez noter pour la colonne 'Opérateur IL' : seule la ligne dans laquelle l'opérateur est utilisé est affichée. On suppose un chargement réussi des (premiers) opérandes nécessaires (p.ex. LD in) dans une des lignes précédentes.

La colonne '**Mod. IL**' indique les différents modificateurs dans l'éditeur IL :

C l'instruction est exécutée uniquement si l'expression précédente fournit le résultat TRUE.

N pour JMPC, CALC, RETC : l'instruction est exécutée uniquement si l'expression précédente fournit le résultat FALSE.

N autres: négation de l'opérande (mais pas de l'accumulateur)

(Une mise entre parenthèses restreint l'opérateur, et l'opération comprise entre les parenthèses n'est exécutée que lorsque la deuxième parenthèse est atteinte

Pour une description détaillée relative à l'utilisation, reportez-vous aux annexes correspondantes des opérateurs dans CoDeSys ou dans les bibliothèques.

Opérateurs dans CoDeSys :

en langage ST :	*en langage IL :*	*Mod. IL*	*Signification*
'			Délimitation d'une chaîne de caractères (p.ex. 'chaîne1')
.. **[ ]**			Tableau : représentation du domaine du tableau (p.ex. ARRAY[0..3] OF INT)
:			Caractère de séparation entre l'opérande et le type dans la déclaration (p.ex. var1 : INT;)Marque la fin d'une commande (p.ex. a:=var1;)
;			Marque la fin d'une instruction (p.ex. a:=var1;)
^			Pointeur déréférencé (p.ex. pointeur1^)
	LD var1	N	Charger la valeur de var1 dans l'accumulateur
:=	**ST var1**	N	Mémoriser le résultat courant à l'emplacement de l'opérande var1
	S boolvar		Positionner l'opérande booléen boolvar sur TRUE et seulement si le résultat courant est TRUE
	R boolvar		Remettre l'opérande booléenne sur FALSE seulement si le résultat courant est TRUE

	JMP marke	**CN**	Saut vers l'étiquette
<Nom du programme>	**CAL** prog1	**CN**	Appeler le programme prog1
<Nom d'instance>	**CAL** inst1	**CN**	Appel du bloc fonctionnel inst1
en langage ST :	*en langage IL :*	*Mod. IL*	*Signification*
<Nom de fonction>(vx, vy,...)>	**<Nom de fonction> vx, vy**	**CN**	Appeler une fonction et passer les variables vx, vy
RETURN	**RET**	CN	Quitter le module et retourner le cas échéant vers l'appelant
	(		La valeur qui suit la parenthèse est considérée comme opérande, l'opération précédente est différée jusqu'à la fin de la parenthèse
	)		Evaluation d'une opération différée
AND	**AND**	N,(	AND bit à bit
OR	**OR**	N,(	OR bit à bit
XOR	**XOR**	N,(	OR exclusif bit à bit
NOT	**NOT**		NOT bit à bit
+	**ADD**	(	Addition
-	**SUB**	(	Soustraction
*****	**MUL**	(	Multiplication
/	**DIV**	(	Division
>	**GT**	(	Plus grand
>=	**GE**	(	Plus grand ou égal
=	**EQ**	(	Égal
<>	**NE**	(	Différent
<=	**LE**	(	Plus petit ou égal
<	**LT**	(	Plus petit
MOD(in)	**MOD**		Division Modulo
INDEXOF(in)	**INDEXOF**		Index interne d'un module in1; [INT]
SIZEOF(in)	**SIZEOF**		Nombre d'octets requis pour le type de données indiqué de in
SHL(K,in)	**SHL**		Décalage bit à bit vers la gauche de K positions d'un opérande
SHR(K,in)	**SHR**		Décalage bit à bit vers la droite de K positions d'un opérande
ROL(K,in)	**ROL**		Rotation bit à bit vers la gauche de K positions d'un opérande

ROR(K,in)	**ROR**		Rotation bit à bit vers la droite de K positions d'un opérande
SEL(G,in0,in1)	**SEL**		Sélection binaire entre 2 opérandes in0 (G est FALSE) et in1 (G est TRUE)
MAX(in0,in1)	**MAX**		Fournit la plus grande de deux valeurs
MIN(in0,in1)	**MIN**		Fournit la plus petite de deux valeurs
LIMIT(MIN,in,Max)	**LIMIT**		Limitation de la plage des valeurs (in est ramené à Min ou à Max en cas de dépassement)
MUX(K,in0,...in_n)	**MUX**		Sélection de la K-ième valeur hors d'une liste de valeurs (in0 à in_n)
ADR(in)	**ADR**		Adresse de l'opérande in [DWORD]
BOOL_TO_<type>(in)	**BOOL_TO_<type>**		Conversion du type de l'opérande booléen en un autre type élémentaire
en langage ST :	*en langage IL :*	*Mod. IL*	*Signification*
<type>_TO_BOOL(in)	**<type>_TO_BOOL**		Conversion du type de l'opérande en BOOLéen
INT_TO_<type>(in)	**INT_TO_<type>**		Conversion du type de l'opérande INT en un autre type élémentaire
REAL_TO_<type>(in)	**REAL_TO_<type>**		Conversion du type de l'opérande REAL en un autre type élémentaire
LREAL_TO_<type>(in)	**LREAL_TO_<type>**		Conversion du type de l'opérande LREAL en un autre type élémentaire
TIME_TO_<type>(in)	**TIME_TO_<type>**		Conversion du type de l'opérande TIME en un autre type élémentaire
TOD_TO_<type>(in)	**TOD_TO__<type>**		Conversion du type de l'opérande TOD en un autre type élémentaire
DATE_TO_<type>(in)	**DATE_TO_<type>**		Conversion du type de l'opérande en un autre type élémentaire
DT_TO_<type>(in)	**DT_TO_<type>**		Conversion du type de l'opérande en un autre type élémentaire
STRING_TO_<type>(in)	**STRING_TO_<type>**		Conversion du type de l'opérande en un autre type élémentaire, in devant avoir une valeur correcte pour le type cible
TRUNC(in)	**TRUNC**		Conversion de REAL en INT
ABS(in)	**ABS**		Valeur absolue de l'opérande
SQRT(in)	**SQRT**		Racine carrée de l'opérande
LN(in)	**LN**		Logarithme naturel d'une opérande

LOG(in)	**LOG**	Logarithme décimal de l'opérande
EXP(in)	**EXP**	Fonction exponentielle de l'opérande
SIN(in)	**SIN**	Sinus de l'opérande
COS(in)	**COS**	Cosinus de l'opérande
TAN(in)	**TAN**	Tangente de l'opérande
ASIN(in)	**ASIN**	Arcsinus de l'opérande
ACOS(in)	**ACOS**	Arccosinus de l'opérande
ATAN(in)	**ATAN**	Arctangente de l'opérande
EXPT(in,expt)	**EXPT** expt	Exponentation de l'opérande à la $expt^{ème}$ puissance

Modules de la bibliothèque Standard.lib :

en langage ST :	*en langage IL :*	*Signification*
LEN(in)	**LEN**	Longueur de la chaîne de caractères de l'opérande
LEFT(str,size)	**LEFT**	Extrait de chaîne situé à l'extrême gauche de la chaîne str (d'une taille size)
RIGHT(str,size)	**RIGHT**	Extrait de chaîne situé à l'extrême droite de la chaîne str (d'une taille size)
MID(str,size)	**MID**	Extrait de chaîne de la chaîne str (taille size)
CONCAT('str1','str2')	**CONCAT** 'str2'	Concaténation de deux chaînes de caractères
INSERT('str1','str2',pos)	**INSERT** 'str2',p	Insertion de la chaîne str1 dans la chaîne str2 à la position pos
DELETE('str1',len,pos)	**DELETE** len,pos	Effacer un extrait de la chaîne de caractères str1 à partir de la position pos, sur une longueur len
REPLACE('str1','str2', len,pos)	**REPLACE** 'str2',len,pos	Remplacer un extrait de la chaîne de caractères str1 à partir de la position pos, sur une longueur len par le contenu de la chaîne de caractères str2
FIND('str1','str2')	**FIND** 'str2'	Recherche d'un extrait de chaîne de caractères str2 dans str1
SR	**SR**	Bloc fonctionnel bistable forcé dominant
RS	**RS**	Bloc fonctionnel bistable réinitialisé
SEMA	**SEMA**	Bloc fonctionnel : sémaphore (interruptible)
R_TRIG	**R_TRIG**	Bloc fonctionnel : front montant reconnu
F_TRIG	**F_TRIG**	Bloc fonctionnel : front descendant reconnu
CTU	**CTU**	Bloc fonctionnel : compteur (comptage)

CTD	**CTD**	Bloc fonctionnel : compteur (décomptage)
CTUD	**CTUD**	Bloc fonctionnel : compteur (comptage-décomptage)
TP	**TP**	Bloc fonctionnel : émetteur d'impulsions
TON	**TON**	Bloc fonctionnel : temporisation à l'enclenchement
TOF	**TOF**	Bloc fonctionnel : temporisation au déclenchement
RTC	**RTC**	Bloc fonctionnel : horloge du système d'exécution

Modules de la bibliothèque Util.lib :

	Signification
BCD_TO_INT	Conversion d'un octet du format BCD en format INT
INT_TO_BCD	Conversion d'un octet du format INT en format BCD
EXTRACT(in,n)	Le n-ième bit de DWORD est donné en format BOOLéen
PACK	Jusqu'à huit bits sont comprimés dans un octet
PUTBIT	un bit dans un DWORD est mis à la valeur existante
UNPACK	un octet est transformé en bits individuels
DERIVATIVE	Dérivation
INTEGRAL	Intégrale
STATISTICS_INT	Valeurs min, max et moyennes en format INT
STATISTICS_REAL	Valeurs min, max et moyennes en format REAL
VARIANCE	Calcul de variance
PD	Régulateurs PD
PID	Régulateurs PID
BLINK	Signal pulsatoire
GEN	Fonctions périodiques
CHARCURVE	Fonctions linéaires
RAMP_INT	Limitation de montée / descente d'une fonction (INT)
RAMP_REAL	Limitation de montée / descente d'une fonction (REAL)
HYSTERESIS	Hystérésis
LIMITALARM	Le dépassement de la limite d'une valeur saisie est vérifié

Appendice F Instructions de ligne et de fichier de commande

9.12 Instructions de ligne de commande

Lors du démarrage de CoDeSys, vous avez la possibilité d'entrer un certain nombre d'instructions qui seront effectuées lors de l'exécution. Ces instructions de ligne de commande commencent par "/". L'écriture en majuscules ou en minuscules n'est pas prise en compte. L'exécution se fait de manière séquentielle, de la gauche vers la droite.

/online	CoDeSys tente de basculer avec le projet actuel vers le mode En Ligne, après le démarrage.
/run	Lancement du programme utilisateur après avoir accédéà CoDeSys. Uniquement valable avec la commande /online.
/show ...	La visualisation de la fenêtre cadre de CoDeSys peut être définie.
	La fenêtre n'est pas visualisée et n'apparaît pas non plus dans la liste des tâches.
/show hide	La fenêtre est affichée de manière réduite.
	La fenêtre est affichée de manière agrandie.
/show icon	La fenêtre est visualisée dans l'état qui était le sien lorsqu'elle a été fermée pour la dernière fois.
/show max	
/show normal	
/out <outfile>	Tous les messages sont écrits en sortie dans le fichier <outfile>, tout en étant affichés dans la fenêtre de messages.
/noinfo	Aucun écran de démarrage n'apparaît au démarrage de CoDeSys.
/userlevel <group>	Le niveau d'accès peut être défini (p.ex. "/userlevel 0" pour le niveau d'accès 0)
/password <password>	Le mot de passe pour le niveau d'accès peut être saisi directement. (p.ex. "/password abc")
/openfromplc	Le projet se trouvant actuellement sur l'automate programmable connecté est chargé.
/visudownload	Lorsque CoDeSys HMI est démarré en même temps qu'un projet qui ne coïncide pas avec celui se trouvant sur l'automate programmable, un téléchargement peut être effectué. (dialogue de confirmation auquel il faut répondre par OUI ou NON)
/cmd <cmdfile>	Les instructions dansle fichier de commande <cmdfile> sont exécutées après le démarrage.

La saisie d'une ligne de commande est structurée comme suit :

"<Chemin du fichier exe CoDeSys>" "<Chemin du projet>" /<Commande1> /<Commande2> ....

Exemple de ligne de commande:

"D:\dir1\codesys" "C:\projects\ampel.pro" /show hide /cmd command.cmd

Le fichier signal.pro s'ouvre sans que la fenêtre ne s'affiche. Le contenu du fichier de commande (cmdfile) command.cmd est exécuté.

9.13 Instructions de fichier de commande (Cmdfile)

Vous trouverez ci-dessous la liste des instructions pouvant être utilisées dans un fichier de commande (<cmdfile>). Et ce fichier peut être appelé via la ligne de commande. L'écriture en majuscules ou en minuscules n'est pas prise en compte. La ligne de commande est affichée sous forme de message dans la fenêtre de messages et écrite dans le fichier de messages (voir ci-dessous), sauf si le caractère "@" est inséré devant l'instruction.

Tous les caractères qui suivent un point-virgule (;) sont ignorés (commentaire). Si les paramètres contiennent des espaces, ils doivent être placés entre guillemets. Les inflexions ne peuvent être utilisées que si le fichier de commande a été créé en code ANSI. Lors de la spécification des paramètres de commande, vous pouvez utiliser des mots-clés. Vous en trouverez une liste à la suite des descriptions de commandes ci-dessous.

Instructions pour contrôler les commandes suivantes

onerror continue	Les commandes suivantes sont exécutées, même si une erreur sest produite.
onerror break	Les commandes suivantes ne pas sont exécutées, même si une erreur sest produite.

Instructions du menu En Ligne:

online login	Accéder au système avec le projet chargé ('En Ligne' 'Accéder au système').
online logout	Quitter le système ('En Ligne' 'Quitter le système')
online run	Démarrer le programme utilisateur ('En Ligne' 'Démarrer')
online sim	Activer la simulation ('En Ligne' 'Simulation')
online sim off	Désactiver la simulation. ('En Ligne' 'Simulation')
online bootproject	Créer un projet d'initialisation'. Regardez la description du commande 'En Ligne' 'Créer projet d'initialisation' !
online sourcecodedownload	Transfert du code source du projet actuel sur l'automate programmable. ('En Ligne' 'Charger code source')
online stop	Arrêt du programme actuel sur le système cible ('En ligne' 'Stop')

Instructions du menu Fichier:

file new	Création d'un nouveau projet ('Fichier' 'Nouveau')
file open <projectfile> mögliche Zusatzbefehle:	Chargement du projet spécifié ('Fichier' 'Ouvrir')
/readpwd:<readpassword>	Le mot de passe pour l'accès en lecture est donné, de sorte que le dialogue de saisie de mot de passe n'apparaisse plus automatiquement avec les projets protégés par mot de passe.
/writepwd:<writepassword>	Le mot de passe pour l'accès complet est donné, de sorte que le dialogue de saisie de mot de passe n'apparaisse plus.
file close	Fermeture du projet chargé ('Fichier' 'Fermer')
file save	Enregistrement du projet chargé ('Fichier' 'Enregistrer')
file saveas <projectfile> Peur être complété en option par : **<type><version>**	Enregistrement du projet chargé sous le nom spécifié ('Fichier' 'Enregistrer sous') Valeur par défaut : Le projet est enregistré comme fichier *.pro dans la version de produit actuelle (version à la création). Si le projet doit

	être enregistré comme bibliothèque interne ou externe, ou encore comme projet sous une version précédente, cela peut être ajouté ici. Entrées possibles pour <type>: "internallib" Sauvegarde comme bibliothèque interne "externallib" Sauvegarde comme bibliothèque externe "pro" Sauvegarde comme projet Entrées possibles pour <version>: 15, 20, 21, 22 (versions de produit 1.5, 2.0, 2.1, 2.2) Exemple : "file save as lib_xy internallib22" -> Le projet xy.pro créé avec la version actuelle de CoDeSys sera enregistré comme lib_xy.lib pour la version 2.2.
file archive <filename>	Le projet actuel complet est archivé dans un fichier ZIP avec comme nom <filename>. ('Fichier' 'Enregistrer l'archive/Envoyer')
file printersetup <filename>.dfr Peut être complété en option par : **pageperobject** ou **pagepersubobject**	Réglages pour la documentation du projet : Fichier cadre *.dfr et entrée optionnelle de disposition des pages pour l'impression: 'pageperobject' (nouvelle page par objet) ou 'page persubobject' (nouvelle page par sous-objet); voir plus bas "project documentation"
file quit	CoDeSys est terminé ('Fichier' 'Quitter')

Instructions du menu Projet:

project build ou **(avant le Service Pack 3 de V2.2)** **project compile**	Le projet chargé est compilé de manière incrémentale ("Projet" + "Compiler")
project rebuild ou **(à partir du Service Pack 3 de V2.2)** **project compile**	Le projet chargé est complètement compilé ("Projet" + "Compiler")
project clean	Les informations de compilation et sur les changements En ligne dans le projet en cours sont effacées('Projet' 'Réorganiser tout')
project import <file1> ... <fileN>	Importation dans le projet chargé des fichiers <file1> & <fileN> spécifiés ('Projet' 'Importer'). Des espaces réservés peuvent être utilisés, p.ex. "project import C:\projects*.exp" importe tous les fichiers avec l'extension *.exp qui se trouvent dans le répertoire C:\projects.
project export <expfile>	Exportation du projet chargé dans le fichier spécifié <expfile> ('Projet' 'Exporter')
project expmul	Exportation de chaque objet appartenant au projet chargé dans un fichier spécifique portant le nom de l'objet.
project documentation	Le projet est imprimé selon les réglages actuels ('Fichier' 'Configuration documentation' ('Projet' 'Documentation du projet'); voir également plus haut "file printersetup")

Instructions pour contrôler le fichier de messages:

out open <msgfile>	Ouverture du fichier spécifié pour la sortie de messages. Ajout de nouveaux messages.
out close	Fermeture du fichier de messages ouvert actuellement
out clear	Effacement de tous les messages contenus dans le fichier de

	messages ouvert actuellement

Instructions pour contrôler les sorties de messages:

echo on	Les lignes de commande sont également affichées sous forme de message
echo off	Les lignes de commande ne sont pas affichées sous forme de message
echo <texte>	Le <texte> est sorti sous forme de message

Instructions pour contrôler le remplacement d'objets ou de fichiers lors de l'importation, l'exportation, le remplacement:

Jusquà V2.2, SP5

replace ok **replace yes**	Remplacer
replace no	Ne pas remplacer
replace yesall	Remplacer tout
replace noall	Ne rien remplacer

à partir de V2.3 :

replace yesall	Remplacer tout (Il nest pas tenu compte dun 'query on' éventuellement positionné, pas de demande de confirmation)
replace noall	Ne rien remplacer (Il nest pas tenu compte dun 'query on' éventuellement positionné, pas de demande de confirmation)
replace query	Si un 'query on' est positionné, une demande de confirmation quant à lobjet à remplacer saffiche, même si 'replace yesall' ou 'replace noall' ont été positionnés

Instructions pour contrôler le comportement par défaut des boîtes de dialogue CoDeSys:

query on	Affichage de boîtes de dialogue destinées à être complétées par l'utilisateur
query off ok	Traitement des boîtes de dialogue comme si l'utilisateur avait cliqué sur OK
query off no	Traitement des boîtes de dialogue comme si l'utilisateur avait cliqué sur Non
query off cancel	Traitement des boîtes de dialogue comme si l'utilisateur avait cliqué sur Annuler

Instruction de débogage:

debug	Correspond à l'instruction /debug dans la ligne de commande

Instruction pour l'appel de fichiers de commande sous forme de sous-programmes:

call <paramètre1> ... <paramètre10>	Appel de fichiers de commande sous forme de sous-programmes. 10 paramètres au maximum peuvent être passés. A l'intérieur du fichier appelé, il est possible d'accéder aux paramètres au moyen de $0 - $9.

Instructions pour définir les répertoires utilisés par CoDeSys:

dir lib <libdir>	Définition de <libdir> comme répertoire des bibliothèques
dir compile <compiledir>	Définition de <compiledir> comme répertoire pour les fichiers de compilation
dir config <config>	Définition de <configdir> comme répertoire pour les fichiers de configuration
dir upload <uploaddir>	Définition de <uploaddir> comme répertoire pour les fichiers de chargement

Instruction pour retarder l'exécution de CMDFILE:

delay 5000	Attendre 5 secondes

Instructions pour contrôler le gestionnaire d'espion et de recettes:

watchlist load <file>	Charger la liste d'espion enregistrée dans <file> et ouvrir la fenêtre correspondante ('Extras' 'Charger la liste d'espion')
watchlist save <file>	Enregistrer la liste d'espion actuelle dans <file> ('Extras' 'Enregistrer la liste d'espion')
watchlist set <texte>	La liste d'espion est définie comme liste d'espion actuelle (correspond à la sélection de la liste dans la partie gauche du gestionnaire d'espion et de recettes)
watchlist read	Actualise les valeurs des variables d'espion ('Extras' 'Lire la recette')
watchlist write	Affecte les valeurs contenues dans la liste d'espion aux variables d'espion ('Extras' 'Ecrire la recette')

Intégration de bibliothèques :

library add <Fichier de bibliothèque1> <Fichier de bibliothèque2> .. <Fichier de bibliothèqueN>	Rattache les fichiers de bibliothèque indiqués à la liste de bibliothèques du projet en cours. Si le chemin daccès au fichier est relatif, le répertoire de bibliothèques défini au sein du projet est employé comme racine du chemin.
library delete [<Bibliothèque1> <Bibliothèque2> .. <BibliothèqueN>]	Efface les bibliothèques mentionnées ou, si aucune bibliothèque nest mentionnée, toutes les bibliothèques contenues dans la liste du projet en cours.

Copie dobjets :

object copy <Fichier projet source> <Chemin source> <Chemin cible>	Copie les objets du chemin source du fichier source indiqué vers le chemin cible du projet qui vient dêtre ouvert. Si le chemin source est le nom dun objet, ce dernier est copié. Sil sagit dun répertoire, tous les objets contenus dans ce répertoire sont copiés. Dans ce cas, la structure sous le répertoire source est également reprise. Si le chemin cible n'existe pas encore, celui-ci est créé.
setreadonly	Avec la valeur TRUE, l'objet n'est accessible qu'en lecture. Pour les

<TRUE|FALSE> <Type d'objet>| <Nom d'objet>	types d'objets pou, dut, gvl, vis, il faut en outre indiquer le nom de l'objet. Types d'objets possibles : pou (module), dut (type de données), gvl (liste des variables globales), vis (visualisation), cnc (objet CNC), liblist (bibliothèques, réglages cible (configuration du système cible), toolinstanceobject (instance dans les outils), toolmanagerobject (toutes les instances dans les outils), customplconfig (configuration de l'automate), projectinfo (informations sur le projet), taskconfig (configuration des tâches), trace, watchentrylist (gestionnaire d'espion et de recettes), alarmconfig (configuration d'alarme) P.ex. seul un accès en lecture est possible après "object setreadonly TRUE pou plc_prg" dans PLC_PRG

Configuration des paramètres de communication (passerelle, appareil) :

gateway local	Définit la passerelle de lordinateur local comme passerelle actuelle.
gateway tcpip <Adresse> <Port>	Définit la passerelle de lordinateur distant indiqué comme passerelle actuelle. <Adresse>: adresse TCP/IP ou nom de lordinateur distant <Port>: port TCP/IP de la passerelle distante Attention : on ne peut atteindre que les passerelles pour lesquelles un mot de passe na pas été défini !
device guid <guid>	Définit lappareil avec le GUID indiqué comme appareil actuel. Le GUID doit correspondre au format suivant : {01234567-0123-0123-0123-0123456789ABC} Les accolades et les traits dunion doivent se trouver aux positions indiquées.
device name<nom dappareil>	Définit lappareil dont le nom est indiqué comme appareil actuel.
device instance <nom dinstance>	Définit le nom dinstance pour lappareil actuel selon le nom indiqué.
device parameter <Id>|<nom de paramètre> <valeur>	Attribue au paramètre dont lidentité ou le nom est indiqué(e) la valeur indiquée, qui est alors interprétée par lappareil.

Appel du système :

system <commande>	Exécute la commande du système dexploitation indiqué.

Sélection du système cible :

target <Id>|<nom>	Définit la plate-forme cible pour le projet en cours. Entrée de lidentité ou du nom tels que définis dans le fichier cible.

Commandes relatives à la gestion du projet dans la base de données de projet ENI :

Dans la description des commandes ci-après, les espaces réservés suivants sont utilisés :

<catégorie>: À remplacer par 'project', 'shared' ou 'compile' pour les catégories de base de données Projet, Objets communs, Fichiers de compilation

<Nom de module> : Le nom de lobjet coïncidant avec le nom dobjet utilisé dans CoDeSys.

<Type dobjet> : À remplacer par labréviation qui a été attribuée à lobjet dans la base de données, servant dextension au nom du module et caractérisant le type dobjet (est définie par la liste des types dobjets, voir Administration ENI, 'Object Types').

Exemple : Objet "GLOBAL_1.GVL" -> le nom du module est "GLOBAL_1", le type dobjet est "GVL" (liste de variables globales)

<commentaire>: À remplacer par le texte de commentaire compris entre guillemets (") qui sera repris dans lhistorique de la version de chaque action.

Commandes pour la configuration de la liaison à la base de données du projet via le serveur ENI :

eni on **eni off**	Loption 'Utiliser base de données projet ENI' est (dés)activée. (Boîte de dialogue 'Projet' 'Options' 'Base de données du projet')
eni project readonly on **eni project readonly off**	Loption 'Accès en lecture seule' pour la base de données Catégorie projet est (dés)activée (Boîte de dialogue 'Projet' 'Options' 'Base de données du projet' 'Objets du projet')
eni shared readonly on **eni shared readonly off**	Loption 'Accès en lecture seule' pour la base de données Catégorie Objets communs est (dés)activée (Dialogue 'Projet' 'Options' 'Base de données du projet', 'Objets communs')
eni set local <nom du module>	Classe le module dans la catégorie 'Local', il nest alors pas géré dans la base de données du projet (Dialogue 'Projet' 'Objet' 'Propriétés' 'Liaison à la base de données')
eni set shared <nom du module>	Classe le module dans la catégorie de base de données 'Objets communs' (Dialogue 'Projet' 'Objet' 'Propriétés' 'Liaison à la base de données')
eni set project <nom du module>	Classe le module dans la catégorie de base de données 'Projet' (Dialogue 'Projet' 'Objet' 'Propriétés' 'Liaison à la base de données')
eni <Catégorie> server <Adresse TCP/IP> <Port> <nom du projet> <nom dutilisateur> <mot de passe>	Configure la connexion au serveur ENI pour la catégorie 'Objets de projet' (Dialogue 'Projet' 'Options' 'Base de données du projet'); Exemple : `eni project server localhost 80 batchtest\project EniBatch Batch` (Adresse TCP/IP= localhost, Port = 80, nom du projet = batchtest\project, nom dutilisateur = EniBatch, mot de passe = Batch)
eni compile sym on **eni compile sym off**	Loption 'Créer une information de symboles ASCII' est (dés)activée pour les objets de la catégorie 'Fichiers de compilation' (Boîte de dialogue 'Projet' 'Options' 'Base de données du projet' 'Configuration ENI' pour 'Fichiers de compilation')
eni compile sdb on **eni compile sdb off**	Loption 'Créer une information de symboles binaires' est (dés)activée pour les objets de la catégorie 'Fichiers de compilation' (Boîte de dialogue 'Projet' 'Options' 'Base de données du projet' 'Configuration ENI' pour 'Fichiers de compilation')
eni compile prg on **eni compile prg off**	Loption 'Créer fichier dinitialisation' est (dés)activée pour les objets de la catégorie Fichiers de compilation (Boîte de dialogue 'Projet' 'Options' 'Base de données du projet' 'Configuration ENI' pour 'Fichiers de compilation')

Commandes du menu 'Projet' 'Liaison avec la base de données' pour travailler avec la base de données :

eni set <categorie>	Lobjet est attribué à la catégorie de base de données ('Définir')
'eni set <categorie>set <type dobjet>:<nom du module> <type dobjet>:<nom du module>	Les objets indiqués dans une liste et séparés par des espaces sont attribués à la catégorie de base de données. ('Définition multiple') Exemple: "eni set project pou:as_fub pou:st_prg" (les modules (pou) as_fub und st_prg sont attribués à la catégorie de base de données)
eni <categorie> getall	Tous les objets de la catégorie sont reprises de la base de données du projet ('Dernières Versions')
'eni <categorie>get <type dobjet>:<nom du module> <type dobjet>:<nom du module>	Les objets de la catégorie donnée, indiqués dans une liste et séparés par des espaces, sont reprises de la base de données. ('Dernière version') Exemple: "eni project get pou:as_fub gvl:global_1" (le module as_fub.pou et la liste des variables globale global_1.gvl sont reprises de la base de données)
eni <categorie> checkoutall "<commentaire>"	Tous les objets de la base de données du projet subissent un check-out. La procédure de check-out est accompagnée dun <commentaire>.
'eni <categorie> checkout "<commentaire>" <type dobjet>:<nom du module> <type dobjet>:<nom du module>	Les objets de la catégorie concernée, indiqués dans une liste par type dobjet:nom du module et séparés par des espaces, subissent une check-out hors de la base de données ('Check out'). La procédure de check-out est accompagnée dun <commentaire> dans lhistorique de la version. Exemple: "eni project checkout "pour l'édition de xy" pou:as_fub gvl:global_1" (POU "as_fub" et la liste de variables globale "global_1" subissent un check-out et cette procédure est commentée de "Pour l'édition de xy")
eni <categorie>checkinall "<commentaire>"	Tous les objets du projet gérés dans la base de données du projet subissent un check-in.La procédure de check-in est accompagnée dun <commentaire>.
'eni <categorie> checkin "<commentaire>" <type dobjet>:<nom du module> <type dobjet>:<nom du module>	Les objets indiqués dans une liste par type dobjet:nom du module et séparés par des espaces subissent un check-in dans la base de données du projet. La procédure de check-in est accompagnée dun <commentaire>.

Mots-clés pour les paramètres de commande :

Lors de la spécification des paramètres de commande, vous pouvez utiliser les mots-clés ci-dessous contenus dans des "$".

$PROJECT_NAME$	Nom du projet CoDeSys en cours (nom du fichier sans lextension ".pro", p.ex. "project_2.pro")
$PROJECT_PATH$	Chemin daccès du répertoire dans lequel le fichier de projet CoDeSys se trouve (sans mention du lecteur et sans barre oblique inversée à la fin, p.ex. "projects\sub1").
$PROJECT_DRIVE$	Lecteur sur lequel se trouve le projet CoDeSys en cours. (sans barre oblique inversée à la fin, p.ex. "D:")
$COMPILE_DIR$	Répertoire de compilation du projet CoDeSys en cours (sans mention du lecteur et sans barre oblique inversée à la fin, p.ex. "D:\codesys\compile")
EXE_DIR	Répertoire dans lequel se trouve le fichier exe CoDeSys (avec mention du lecteur et sans barre oblique inversée à la fin, p.ex. D:\codesys)

Exemple de fichier de commande <command file name>.cmd :

```
file open C:\projects\CoDeSys_test\signal.pro
query off ok
watchlist load c:\work\w.wtc
online login
online run
delay 1000
watchlist read
watchlist save $PROJECT_DRIVE$\$PROJECT_PATH$\w_update.wtc
online logout
file close
```

Ce fichier de commande ouvre le fichier de projet signal.pro, charge une liste d'espion chargée sous le nom w.wtc, démarre le programme utilisateur, écrit les valeurs des variables dans la liste d'espion w_update.wtc au bout d'1 seconde et enregistre également la liste d'espion dans le répertoire "C:\projects\CoDeSys_test" avant de refermer le projet.

Le fichier de commande est appelé dans une ligne de commande comme suit :

"<Chemin d'accès Fichier Exe CoDeSys>" /cmd "<Chemin d'accès Fichier cmd>"

Importation de fichiers Siemens

Le sous-menu 'Projet' '**Importer un fichier Siemens**' propose des commandes destinées à l'importation de blocs et de variables contenus dans les fichiers Siemens STEP5 et STEP7. La commande '**Importer un fichier SEQ**' permet d'importer des variables globales contenues dans le fichier de symboles STEP5. Il convient d'activer cette option avant d'appeler les commandes '**Importer un fichier S5**', de façon à ce que des noms symboliques utilisables soit créés à partir d'adresses absolues, lors de l'importation de blocs. Ces deux commandes servent respectivement à importer des blocs à partir de fichiers programme STEP5. Ce faisant, les blocs sont insérés dans le projet **CoDeSys** qui est ouvert. Vous pouvez choisir de laisser les blocs en langage STEP5-IL ou bien de les convertir dans un langage CEI.

Il est préférable que le projet **CoDeSys** vers lequel vous importez soit vide. Il faut toutefois veiller à ce que la bibliothèque standard.lib soit liée à votre projet, sinon vous ne pourrez importer ni compteurs ni temporisateurs.

Importer un fichier SEQ

Le format SEQ est un format courant pour le fichier symbolique dans un projet STEP5 ou STEP7. Il est possible de lire des affectations symboliques à partir du fichier symbolique SEQ (*.seq). Une affectation symbolique contient une adresse absolue d'un élément de programmation S5(entrée, sortie, mémento, etc.), l'identificateur symbolique correspondant et, en option, un commentaire sur le symbole. Un fichier SEQ est un fichier texte qui contient une affectation de ce type par lignes. Les "champs" de l'affectation sont séparés par des tabulations. Une ligne peut ne comporter qu'un seul commentaire; dans ce cas, elle doit commencer par un point-virgule.

Les affectations symboliques dans un fichier SEQ sont compilées en déclarations de variables globales, d'après la norme CEI 61131-3. Ce faisant, le nom symbolique, l'adresse et, le cas échéant, le commentaire sont pris en compte. L'adresse est adaptée d'après la norme CEI 61131-3 (signe de pourcentage etc.). Etant donné qu'un nom symbolique S5 peut contenir des caractères qui ne sont pas autorisés au sein d'un identificateur CEI, il se peut que le nom soit modifié. Les caractères non valides sont tout d'abord remplacés par des caractères de soulignement; pour éviter que plusieurs caractères de soulignement ne se suivent directement, le deuxième caractère de soulignement est remplacé dans ce cas par un caractère valide (p.ex. '0'). Si un nom symbolique a été modifié en cours de conversion, alors le nom d'origine est rajouté derrière le nouveau nom sous forme de commentaire. Les lignes de commentaire SEQ sont prises en compte à titre de commentaires. Il est possible de créer plusieurs blocs de variables globales. Chaque bloc comprend moins de 64K de texte.

Le format SEQ, décrit précédemment, est utilisé par le logiciel Siemens STEP5 PG et par l'ACCON PG de DELTALOGIC. Ce format SEQ ne contient aucun caractère de séparation (tabulations), mais table au contraire sur une longueur fixe des noms symboliques et complète si nécessaire les noms au moyen d'espaces.

L'utilisateur choisit dans une boîte de dialogue Windows standard le fichier SEQ. Ensuite l'importation est effectuée, et pour finir la liste de variables globale est compilée. Ce faisant, des erreurs peuvent se produire, à cause de la transformation d'identificateurs STEP5 en identificateurs conformes à la norme CEI 61131-3. Prenons un exemple. Deux identificateurs STEP5 "A!" et "A?" sont convertis tous deux en identificateur CEI "A_", ce qui provoque l'affichage du message "Plusieurs déclarations possèdent des identificateurs identiques 'A_'" Modifiez un des deux identificateurs.

Il ne faut en aucun cas effectuer des modifications autres que celle-ci dans la liste de variables globales. Si vous voyez des adresses qui sont valides sur un automate programmable Siemens mais pas sur votre propre automate, laissez les provisoirement telles quelles, même si des milliers de messages d'erreur apparaissent lors de la compilation. Ces adresses seront utilisées telles quelles lors de l'importation des blocs.

Si le projet vers lequel vous importez contient déjà une déclaration de variable globale x avec une adresse (p.ex. "%MX4.0"), il se peut qu'une nouvelle variable avec la même adresse soit définie, au cours de l'importation SEQ. La norme CEI 61131-3 autorise cela, mais souvent ce n'est pas vrai pour l'utilisateur. Vous n'obtenez aucun message d'erreur, mais votre programme ne fonctionnera peut être pas comme souhaité, car l'adresse est utilisée dans plusieurs modules sans aucun lien entre eux. C'est pourquoi il est préférable d'importer vers un projet vide ou vers un projet qui n'utilise pas (encore) d'adresses absolues.

Après avoir effectué l'importation SEQ, vous pouvez à présent importer des blocs STEP5 ou 7. Vous pouvez aussi dès à présent insérer les entrées et sorties utilisées dans la configuration de l'automate. Celles-ci ne sont pas requises pour l'importation STEP5, mais les adresses utilisées sont vérifiées dès que vous compilez à nouveau votre projet, et sont signalées comme erreurs, le cas échéant.

Importer un fichier S5

Il est possible de lire des blocs à partir de fichiers programme Siemens S5 (*.s5d). Le code contenu est le code MC-5, qui peut être exécuté par l'automate programmable S5. Le code MC5 correspond en général directement à la liste d'instructions STEP5 (sans noms symboliques). De plus, le SD5 contient les commentaires relatifs aux lignes de la liste d'instruction STEP5. Etant donné qu'un fichier S5D ne contient aucun nom symbolique, mais uniquement des adresses absolues, **CoDeSys** cherche le nom symbolique correspondant à l'adresse dans les variables existant déjà au sein du projet **CoDeSys**. Si **CoDeSys** ne trouve aucun nom, alors l'adresse absolue reste inchangée. Par conséquent, si vous attachez de l'importance aux noms symboliques, importez le fichier SEQ avant d'importer le fichier S5.

L'utilisateur choisit dans une boîte de dialogue Windows standard le fichier S5D. Ensuite une autre boîte de dialogue propose la liste des blocs pouvant être sélectionnés. Il est préférable de choisir la totalité des blocs. Vous pouvez également choisir de laisser les blocs en langage STEP5-IL ou de les convertir en langage IL, LD ou FBD.

Dans la mesure du possible, on utilise, au cours de l'importation, des noms symboliques en lieu et place des adresses absolues. Lorsque **CoDeSys** trouve une instruction telle que "U M12.0" pendant l'importation, alors il cherche une variable globale qui est définie sur le mémento M12.0. La première déclaration qui convient est retenue et l'instruction est importée sous forme de "U –Nom" et non sous la forme "U M12.0" (Nom est ici l'identificateur symbolique du mémento M12.0).

Parfois, l'importation ou la conversion du code nécessite des variables supplémentaires. Celles-ci sont déclarées globalement. Par exemple, pour reproduire des entrées activées par un front montant ou descendant (p.ex. pour un compteur S5), on doit faire appel à des instances R_TRIG.

Conversion de S5 vers CEI 61131-3

Si, pour l'importation STEP5, vous avez choisi un langage CEI comme langage cible, vous devez tabler sur le fait que votre projet n'est pas convertible intégralement d'après la norme CEI 61131-3. Si une partie d'un bloc S5 contient un code qui n'est pas convertible d'après la norme CEI 61131-3, alors un message d'erreur approprié est affiché et le code STEP5 IL d'origine qui pose problème est intégré dans le module CEI sous forme de commentaire. En règle générale, ce code doit être remplacé par vous même, c.-à-d. que vous devez le réécrire. Les commandes système qui ne fonctionnent que sur un système S5 donné ne sont pas convertible en CEI. Le "jeu de commandes de base STEP5" peut être converti en code CEI en appuyant sur un bouton, bien que STEP5 présente des différences de principe considérables.

Le jeu de commandes de base, convertible d'après la norme CEI 61131-3, comprend toutes les commandes d'un système de programmation STEP5 convertibles en langage LD ou FBD, ainsi que toutes les commandes qui sont autorisées dans un PB-STEP5 (bloc de programme). De plus, parmi les commandes STEP5 qui sont autorisées uniquement en langage IL ou dans des FB (blocs fonctionnels), celles qui sont réellement disponibles sur tous les systèmes S5, comme par exemple les sauts absolus ou conditionnels, les commandes de décalage, etc., sont généralement convertibles en CEI.

La seule exception ou restriction au niveau de la conversion concerne la remise à zéro des temporisateurs, qui est possible en STEP5 mais pas avec la norme CEI 61131-3.

Commandes convertibles, dans le détail:

U, UN, O, ON, S, R, = avec les opérandes binaires suivants: E (entrées), A (sorties), M (mémentos), S (mémentos S), D (données dans des blocs de données)

U, UN, O, ON avec les opérandes suivants: T (temporisateurs), Z (compteurs)

S, R avec les opérandes suivants: Z

SU, RU, P, PN avec les opérandes suivants: E, A, M, D

O, O(, U(,)

L, T avec les plages d'opérandes suivantes: E, A, M, D, T, Z, P (périphérie) et les tailles d'opérandes suivantes: B (octet), W (mot), D (mot double), L (octet gauche), R (octet droit)

L avec les formats de constantes suivants: DH, KB, KF, KH, KM, KT, KZ, KY, KG, KC

SI, SE, SA avec les opérandes suivants: T

ZV, ZR avec les opérandes suivants: Z

+, -, X, : avec les opérandes suivants: F (nombre en virgule fixe), G (nombre en virgule flottante)

+, - avec les opérandes suivants: D (nombre en virgule fixe à 32 bits)

!=, ><, >, <, >=, <= avec les opérandes suivants: F, D, G

ADD avec les opérandes suivants: BF, KF, DH

SPA, SPB avec les opérandes suivants: PB, FB (avec la plupart des types de paramètres), SB

A, AX avec les opérandes suivants: DB, DX

BE, BEA, BEB

BLD, NOP, ***

UW, OW, XOW

KEW, KZW, KZD

SLW, SRW, SLD, RRD, RLD

SPA=, SPB=

SPZ=, SPN=, SPP=, SPM=

TAK

D, I

La plupart des commandes d'opérateurs formels

Commandes non convertibles

U, UN, O, ON, S, R, = avec les opérandes binaires suivants: Bits de temporisateurs et de compteurs (T0.0, Z0.0)

L, T avec les plages d'opérandes suivantes: Q (périphérie étendue)

LC avec les opérandes suivants: T, Z

SV, SS, R, FR avec les opérandes suivants: T

FR avec les opérandes suivants: Z

Commandes d'opérateurs formels pour lancer, remettre à zéro et libérer des temporisateurs

Toutes les commandes concernant des opérandes issus des plages BA, BB, BS, BT (données du système d'exploitation).

SPA, SPB avec les opérandes suivants: OB (fonctionne seulement sur certains S5 avec certains OB)

BA, BAB avec les opérandes suivants: FX

E, EX avec les opérandes suivants: DB, DX

STP, STS, STW

DEF, DED, DUF, DUD

SVW, SVD

SPO=, SPS=, SPR

AS, AF, AFS, AFF, BAS, BAF

ENT

SES, SEF

B avec les opérandes suivants: DW, MW, BS

LIR, TIR, LDI, TDI, TNW, TXB, TXW

MAS, MAB, MSA, MSB, MBA, MBS

MBR, ABR

LRW, LRD, TRW, TRD

TSG

LB, TB, LW, TW avec les opérandes suivants: GB, GW, GD, CB, CW, CD

ACR, TSC

BI

SIM, LIM

Si vous considérez les commandes non convertibles, vous constatez que la plupart sont des commandes particulières, qui sont disponibles uniquement sur certains systèmes. Les commandes standard non convertibles sont: le chargement de valeurs de temporisateur ou de compteur codifiés en BCD (LC T, LC Z), les types de temporisateurs SV, SS et la remise à zéro de temporisateurs.

Blocs de données

Les blocs de données STEP5 sont convertis en modules dotés d'un en-tête, mais dépourvus de code. Cela représente un avantage si les blocs de données sont utilisés comme une plage normale de variables et un désavantage si l'on a tenté soi-même d'implémenter des concepts, tels que des blocs de données servant d'instance, dans le programme STEP5.

Autres problèmes pouvant survenir lors de l'importation STEP5

Il est possible d'améliorer soi-même l'importation STEP5 pour les points suivants:

1. Valeurs de temps dans des variables en format mot

Dans STEP5, une valeur de temps peut se trouver dans n'importe quelle adresse WORD, que cela soit à l'intérieur de la plage des mémentos ou dans un bloc de données. Cela n'est pas autorisé d'après la norme CEI 61131-3. Les variables et les constantes TIME ne sont pas compatibles avec les adresses WORD. En cours d'importation STEP5, il se peut qu'une séquence de commandes comportant une erreur de ce type soit générée. Cela ne se produira pas si vous accédez à un bloc de données en ayant sélectionné le format de temps (KT) pour l'adresse correspondante. Par conséquent, l'erreur n'apparaît que lorsque le programme STEP5 est au moins susceptible d'être amélioré. Vous reconnaissez cette erreur grâce au message "Types incompatibles: Ne peut pas convertir WORD en TIME " ou „Types incompatibles: Ne peut pas convertir TIME en WORD". Lors d'un traitement ultérieur vous devrez changer la déclaration de la variable WORD (si elle existe) et en faire une variable TIME.

2. Erreur lors de l'accès à des blocs de données

Les blocs de données ne sont pas reconnus par la norme CEI 61131-3 et il n'est pas possible de les constituer d'après cette norme. Sans STEP5 les blocs de données sont soit utilisés comme plage normale de variables (comme par exemple la plage des mémentos), soit sous forme de tableau (B DW), de pointeur (B MW100, A DB 0) ou d'union (accès aux octets, mots ou mots doubles dans les DB). La conversion STEP5 peut convertir des accès à des DB uniquement si ceux-ci s'effectuent avec un minimum de structuration. Concrètement, il faut savoir quel DB est actuellement ouvert (A DB) pour effectuer un accès. Cette condition est remplie si l'opération A DB est situé plus avant dans le même bloc ou si le numéro de DB est donné avec le bloc sous forme de paramètre formel. Si aucun A DB n'est placé avant le premier accès au DB, alors le bloc ne peut pas être converti. Vous reconnaissez cela grâce au message "Aucun bloc de données ouvert (insérez un A DB)". Au niveau du bloc converti en module vous constatez dans ce cas des accès à des variables non définies, portant par exemple le nom "ErrorDW0", qui génèrent des messages d'erreur lors de la compilation du bloc venant d'être converti en module. Vous pouvez alors remplacer les variables par un accès au DB approprié, par exemple en remplaçant partout "ErrorDW0" par "DB10.DW0". Une autre possibilité consiste à éliminer le bloc converti en module et d'insérer un A DB au début du bloc STEP5.

Dans tous les cas, il faut qu'un bloc STEP5 qui accède à des mots de données (-octets, etc.) commence par ouvrir le bloc de données. Au besoin, il faut que le bloc soit amélioré avant

l'importation, en insérant la commande A DB appropriée, de préférence au début du bloc. Autrement le bloc converti devra faire l'objet d'un traitement ultérieur.

S'il existe plusieurs opérations A DB et qu'on les contourne en partie au moyen de sauts, alors la conversion peut s'avérer erronée du fait qu'un code accédant à un DB impropre est généré.

3. Concepts évolués concernant l'accès à des blocs de données

Dans STEP5, il existe la possibilité de réaliser soi-même quelque chose qui ressemble à une instance, en faisant en sorte qu'un bloc de code ouvre un bloc de données d'après son indexation. C'est ce qui se produit par exemple avec la séquence de code suivante:

```
L KF +5
T MW 44
B MW 44
A DB 0
```

Dans le cas présent, c'est le DB5 qui est finalement ouvert (de façon générale le DB ouvert est celui dont le numéro figure dans le mot de mémento %MW44). De tels accès ne sont pas pris en compte lors de la conversion, par conséquent il faut procéder ultérieurement au traitement suivant:

Il faut d'abord importer tous les DB qui fonctionnent comme instance, comme par exemple le DB5 et le DB6. Vous pouvez choisir si ces dernier seront importés sous forme de modules IL, LD ou FBD classiques. Les modules n'ont pas de code. Il possèdent seulement un en-tête avec des définitions de variables locales. Il est possible de réaliser des instances de type à partir de ces modules. Créez un type défini par l'utilisateur (que vous appelez par exemple DBTyp), dans lequel vous insérez à titre de composant la variable locale du DB converti. Créez ensuite des instances globales avec le type que vous venez de définir, en écrivant dans une liste de variables globales:

```
VAR_GLOBAL
 DB5, DB6 : DBTyp;
END_VAR
```

Vous pouvez maintenant supprimer du projet les DB convertis.

Vous devez ensuite créer l'ouverture des DB d'après leur indexation, en attribuant au bloc correspondant un paramètre VAR_INPUT supplémentaire, de type DBTyp. Les accès aux données au sein du bloc doivent ensuite être modifiés en fonction de cette instance. Au moment de l'appel, il faut transférer comme paramètre actuel un des DB servant comme instance.

4. Les blocs fonctionnels S5 „intégrés", qui possèdent une interface d'appel STEP5, mais pour lesquels l'implémentation n'est pas écrite en STEP5 (ou en MC5, selon le cas) ou pour lesquels un mécanisme de protection particulier est prévu, jouent un rôle particulier. Les blocs de ce type se présentent la plupart du temps sous forme de firmware et „peuvent être importés uniquement sous forme d'interface". Leur partie d'implémentation reste vide. Ces blocs doivent en principe être reprogrammés lors d'une conversion.

5. Il existe aussi des OB de type firmware, qui ne possèdent pas d'interface et dont le code n'est pas en format STEP5, mais par exemple en assembleur 805xx. Cela concerne avant tout le régulateur PID dans sa version OB251, qui reçoit ses paramètres et ses variables locales d'un autre bloc (de données), qui peut être choisi par l'utilisateur. Ni les régulateur PID, ni les blocs de données correspondants, ni aucun autre bloc utilisant le régulateur en accédant sur le bloc de données, ne sont convertibles en CEI. Le code CEI qui est généré, lors de la conversion, pour les blocs de données et les autres blocs, n'est pas utile en l'absence du régulateur PID. La signification des parties de programme peut être obtenue en consultant le manuel de programmation qui se rapporte à l'unité centrale concernée.

6. Pour configurer des unités centrales S5, et parfois des sous-groupes, il existe parfois des blocs de données de configuration, comme le DB1 (S5-95U), DX0, DX2, qui sont convertis en modules CEI sans utilité. La signification de telles données est obtenue d'une part en consultant le manuel de programmation qui se rapporte à l'unité centrale concernée, et d'autre part en faisant appel à un système de programmation S5 capable d'évaluer les DB de configuration. La configuration concerne par exemple les réglages au niveau de la communication, du traitement de valeurs analogiques, des systèmes multiprocesseurs, etc. C'est pourquoi il n'est pas intéressant d'apporter une quelconque

modification à ces blocs lors du transfert vers un automate programmable qui n'est pas fabriqué par Siemens.

A la fin de l'importation vous devez rechercher les erreurs signalées par des messages et effectuer aux endroits correspondants les corrections, les ajouts ou les remplacements nécessaires. Ces endroits sont signalés par un commentaire, qui commence par:

(*ATTENTION! Code STEP5 non convertible affiché en commentaire:*)

Le commentaire contient ensuite le code non convertible.

Pour terminer, il faut prendre en considération les adresses. Lors de l'importation, des adresses Siemens d'origine sont générées. En voici la structure:

Bits: Byte-Offset.Bit-Numéro

Autres que des bits:Byte-Offset

De plus, les adresses WORD qui se suivent immédiatement se recoupent (car les nombres dans les adresses sont des bytes-offsets). C'est pourquoi un byte de %MW32 et un byte de %MW33 se recoupent, en l'occurrence %MB33 (cela n'est vrai que pour un automate programmable Siemens). Sur votre automate programmable, %MW32 et %MW33 sont complètement indépendant l'un de l'autre.

Votre automate programmable possède éventuellement plusieurs niveaux de hiérarchie, les éléments autres que des bits ayant par exemple plusieurs niveaux d'emboîtement ("%MW10.0.0" sous forme de WORD). Vous pouvez, soit compléter les adresses de façon à les rendre compatibles avec votre automate programmable, soit renoncer complètement aux adresses. Dans ce cas, vous devez être très prudent !! Dans les programmes Siemens d'origine, on rencontre souvent, dans la même zone de mémoire, tantôt un accès WORD et tantôt un accès binaire ou byte. Lors de l'importation dans **CoDeSys,** ces accès sont compilés correctement seulement dans le cas de blocs de données. Dans ce cas, **CoDeSys** crée des variables WORD pour les mots à l'intérieur les DB. L'accès WORD au mot X à l'intérieur du DB Y s'effectue alors sans problème. Les accès au byte gauche ou droit du mot X, les accès mot double ou les accès binaires sont compilés sous forme d'expressions complexes. Cela n'est pas possible dans le cas de mémentos, d'entrées ou de sorties, étant donné qu'on ne peut pas tabler sur un type d'accès standard (p.ex. accès WORD). Par conséquent, si le programme travaille tantôt avec %MX33.3, tantôt avec %MB33, %MW32 ou %MD30, alors vous devez convertir cela par vous même, non sans peine. Sinon, le programme CEI généré par **CoDeSys** lors de l'importation comportera à coup sûr des erreurs.

Faites sortir une liste des références croisées pour l'ensemble des entrées, sorties et mémentos, pour détecter les accès problématiques. Ecartez les accès qui interfèrent.

Appendice G Dialogues de la configuration du système cible

9.14 Dialogue de la configuration du système cible

Les configurations des systèmes cibles prédéfinies dans le fichier cible peuvent encore le cas échéant être modifiées par l'utilisateur dans CoDeSys. Il faut pour ce faire qu'ils aient été entrés dans le fichier cible avec les configurations 'visible' et 'modifiable'.

Vous trouverez la boîte de dialogue '**Paramètres de la cible**' à la rubrique du même nom sous l'onglet Ressources, et cette boîte de dialogue affiche au minimum dans le champ derrière **Configuration** le nom de la configuration de la cible concernée. Vous trouverez en outre éventuellement quatre onglets sous lesquels différentes configurations quant à

- Plate-forme de la cible
- Composition du mémoire
- Général
- Fonctions de réseau
- Visualisation

sont rendues visibles et peuvent également être modifiées.

Attention : Veuillez noter que toute modification de la configuration préréglée du fabricant peut avoir de lourdes conséquences sur le comportement du système cible !

Après un changement, vous pouvez revenir aux valeurs de la configuration par défaut par le biais du bouton **Par défaut**.

La description du contenu possible des catégories de boîtes de dialogue Répartition de la mémoire, Généralités et Fonctions réseau valent pour toutes les plates-formes. Les points de la catégorie du dialogue Plate-forme cible varient quant à eux en fonction des différents systèmes cibles.

Les plates-formes suivantes sont actuellement supportées :

- Intel 386 et compatibles
- Motorola 68k
- Infineon C16x
- PowerPC
- Intel StrongARM
- MIPS III ISA
- Hitachi SH
- 8051

9.15 Catégorie: Plate-forme cible

9.15.1 Système cible 'Intel 386 et compatibles', Catégorie Plate-forme de la cible

Boîte de dialogue Paramètres du cible 'Intel 386 et compatibles', Plate-forme de la cible

Paramètres du cible

Configuration: CoDeSys SP for x86 V2.3

Plate-forme de la cible | Composition du mémoire | Général | Fonctions de réseau

Plate-forme: Intel 386 compatible

☑ Processeur virgule flottante

☑ Débogage dans l'environnement multi-tâches

☐ Sauts optimé

☐ Opérations optimées avec des constantes

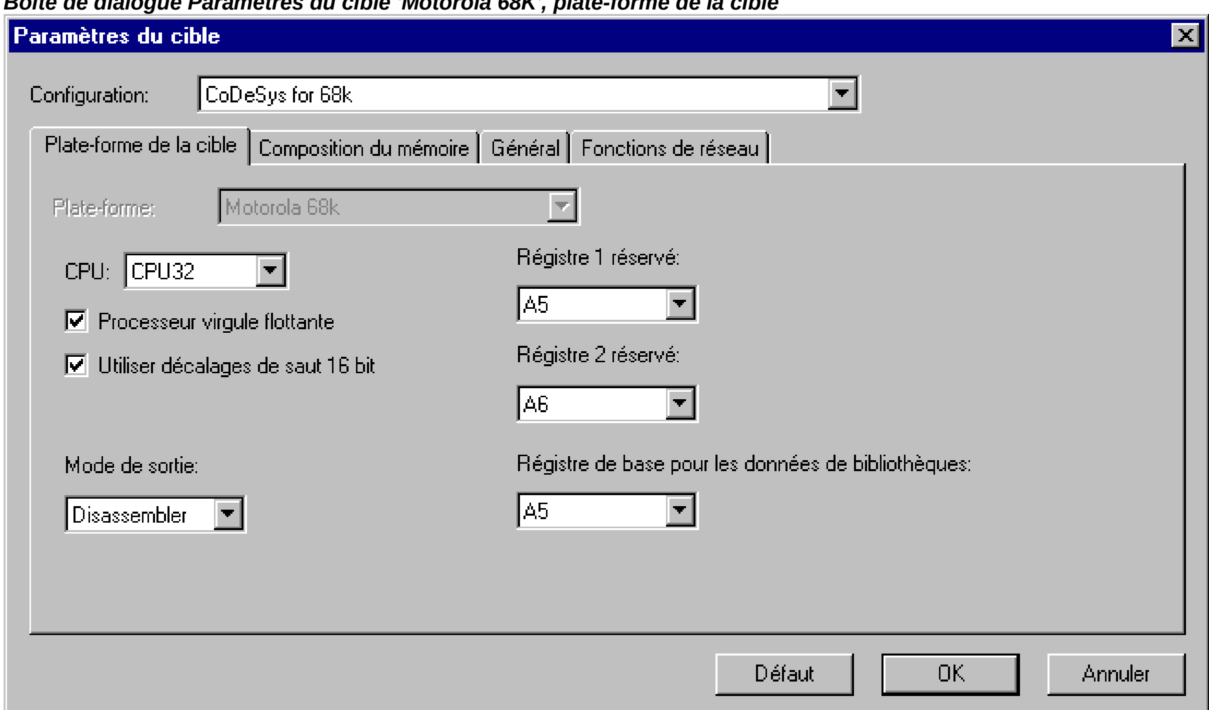
☐ Opérations de chargement optimées

Défaut | OK | Annuler

Point du dialogue	**Signification**
Plate-forme	Type de système cible
Options de processeur à virgule flottante	Les commandes FPU sont générées sur la plate-forme x86 pour les opérations à virgule flottante
Débogage en environnement multitâche	Du code supplémentaire permettant le débogage dans des systèmes multitâche est généré
Sauts optimé	Sauts conditionnels améliorés après comparaison ; Code plus rapide et plus court (particulièrement sur 386/486); Lignes avec conditions de saut apparaissent en gris dans le Flowcontrol
Opérations optimées avec des constantes	Opérations améliorées avec des constantes (A = A + 1, A < 500 etc.); Code plus rapide et plus court (particulièrement sur 386/486); les constantes apparaissent en gris dans le Flowcontrol
Opérations de chargement optimées	Les opérations de chargement sont omises en cas d'accès multiple à une variable / une constante ; Code plus rapide et plus court

9.15.2 Système cible Motorola 68K, Catégorie Plate-forme de la cible

Boîte de dialogue Paramètres du cible 'Motorola 68K', plate-forme de la cible



Point du dialogue	Signification
Plate-forme	Type de système cible
CPU :	Variante du processeur 68k; version de base 68000 ou CPU32 et plus; Valeurs valides: CPU32|68K
Processeur virgule flottante	Les commandes FPU sont générées pour les opérations à virgule flottante
Utiliser décalages de sauts 16 bits	activé : les sauts pour l'évaluation d'expressions booléennes fonctionnent avec des décalages relatifs de16 bits (expressions plus complexes possibles, mais plus de code) ; désactivé: des décalages 8 bits sont utilisés ;
Registre réservé 1 :	Le registre d'adresses indiqué est réservé et n'est pas utilisé. Si "None" est indiqué, il peut être utilisé par le générateur de code; Valeurs valides : None, A2, A4, A5, A6
Registre réservé 2 :	Autre registre d'adresses réservé. Le registre d'adresses indiqué est réservé et n'est pas utilisé. Si "None" est indiqué, il peut être utilisé par le générateur de code; Valeurs valides : None, A2, A4, A5, A6
Registre de base pour les données de bibliothèques :	Registre pour l'adressage de données statiques au sein de bibliothèques C (l'adresse de l'espace libre y est chargée avant l'appel des fonctions de bibliothèque)
Structures alignées sur octet autorisées	Adressage en mode octet (adresse impaires également possibles)
Mode de sortie	Réglage "Assembleur" ou "Désassembleur" : Lors de la compilation, un fichier en format hexadécimal est créé dans le répertoire de compilation configuré ('Projet' 'Options' 'Répertoires'). Celui-ci contient le code Assembleur ainsi que le code Désassembleur qui ont été créés. Réglage "Nothing" : pas de fichier en format hexadécimal créé.

9.15.3 Système cible Infineon C16x, Catégorie Plate-forme de la cible

Boîte de dialogue Paramètres du cible 'Infineon C16x', plate-forme de la cible

Paramètres du cible
Configuration: CoDeSys SP for SAB C16x Development
Plate-forme de la cible | Composition du mémoire | Général | Fonctions de réseau
Plate-forme: Infineon C16x
Code
Compilateur: Keil
Taille de la pile: 32
Données: Huge
Fonctions: Huge
☑ Init. fonctions ☐ Optimer
Sortie
☐ HEX ☐ LST ☐ MAP
☑ LST avec adresses
DPPs
DPP0: None
DPP1: None
DPP2: None
Dans les instances: None
Bibliothèques
Code: 16#5000
Tableaux: 16#0
Données: 16#30000
Longueur des donné: 16#8000
Bloc fonctionnels: 512
Références: 128
Défaut | OK | Annuler

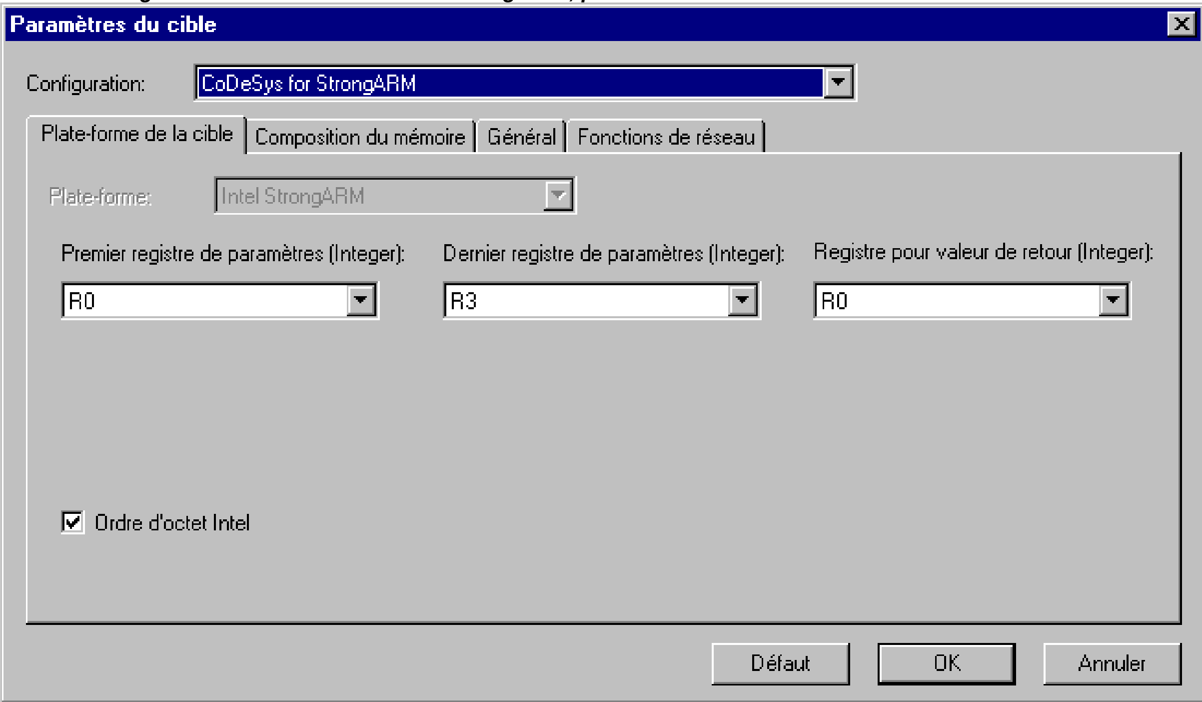
Point du dialogue	**Signification**
Plate-forme	Type de système cible
Code / Compilateur :	Compilateur utilisé lors de la compilation du système cible et des bibliothèques (en raison des conventions d'appel C) BSO-Tasking|Keil
Code / Taille de pile :	Profondeur maximale d'appel (emboîtement) Valeurs valides : 32, 64, 96, 128, 160, 192, 224, 256
Code / Données	Type de mise en mémoire pour les données Valeurs valides : Huge, Far, Near
Code / Fonctions	Type de mise en mémoire pour le code Valeurs valides : Huge, Near
Init. fonctions	activé=les fonctions comportent du code d'initialisation pour les variables locales
Optimer	activé=optimisation du code pour les index de tableaux constants
Sortie HEX	activé=création d'un fichier en format hexadécimal avec le code
Sortie BIN	activé=création d'un fichier binaire avec le code
Sortie MAP	activé=création d'un fichier MAP avec le code
Sortie LST	activé=création d'un fichier de listage avec le code
Sortie LST, avec adresses	activé=création d'une liste des adresses du code
Bibliothèques / **Code** **Tableaux** **Données** **Longueur des donné** **Bloc Fonctionels**	Réglages pour les bibliothèques : Adresse de départ pour le code Adresse de départ pour les tableaux Adresse de départ pour les données Longueur de toutes les données des bibliothèques Nombre maximal de modules de bibliothèque : Valeurs valides : 0-512

Références	Nombre maximal de références
DPPs / DPP0..DPP2	Data Page Pointer 0 bis 2 sont mises Valeurs valides: None, Auto, Page 0, Page 1,...,Page 255
Dans les instances	DPP pour breve adressement dans instances de blocs fonctionnels Valeurs valides: None, DPP0, DPP1, DPP2

9.15.4 Systèmes cibles Intel StrongARM et Power PC, Catégorie Plate-forme de la cible

Les points de dialogue sont identiques pour les deux systèmes cibles.

Boîte de dialogue Paramètres du cible 'Intel StrongARM', plate-forme de la cible

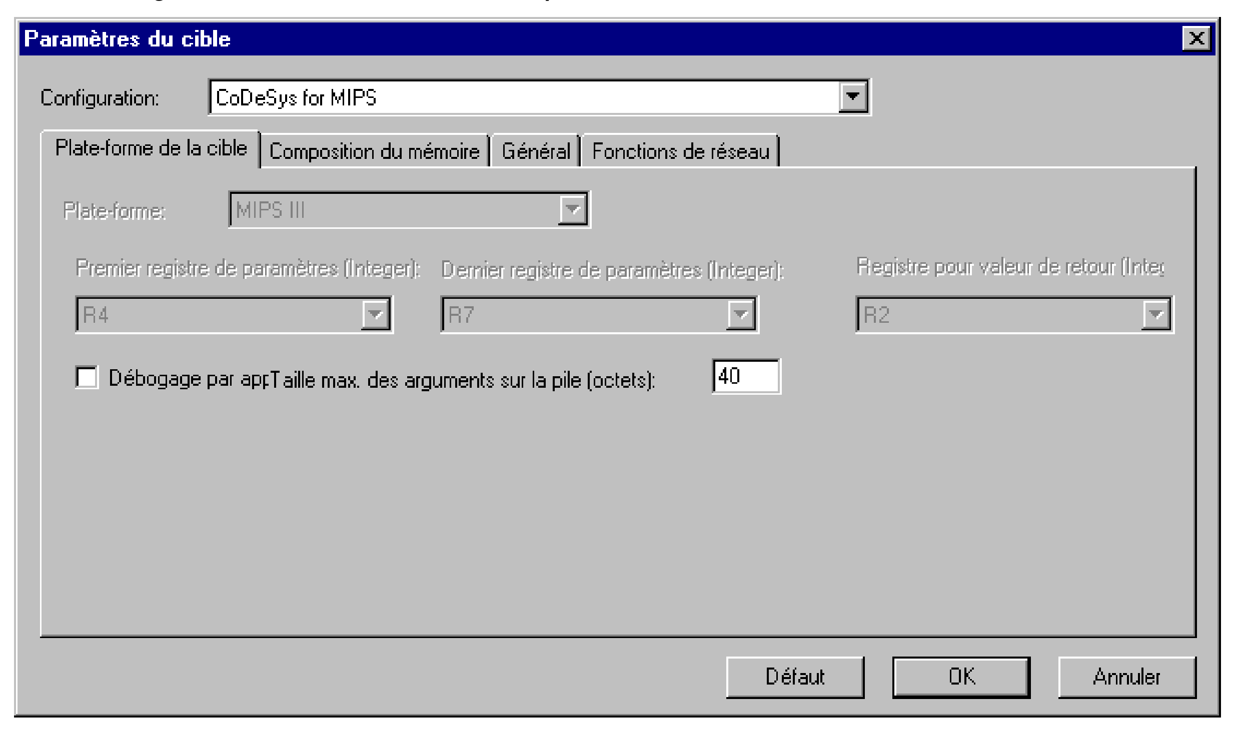


Point du dialogue	Signification
Plate-forme	Type de système cible
Processeur à virgule flottante	activé : Les commandes FPU sont générées pour les opérations à virgule flottante
Premier registre de paramètres (Integer)	Registre dans lequel le premier paramètre (nombre Integer) d'appel de fonction C est transmis (domaine dépendant du système d'exploitation)
Dernier registre de paramètres (Integer)	Registre dans lequel le dernier paramètre (nombre Integer) d'appel de fonction C est transmis (domaine dépendant du système d'exploitation)
Registre pour valeur renvoyée (Integer)	Registre dans lequel les valeurs Integer d'appel de fonction C sont renvoyées (domaine dépendant du système d'exploitation)
Premier registre de paramètre (virgule flottante)	Registre dans lequel le premier paramètre (nombre à virgule flottante) d'appel de fonction C est transmis (domaine dépendant du système d'exploitation)
Dernier registre de paramètre (virgule flottante)	Registre dans lequel le dernier paramètre (nombre à virgule flottante) d'appel de fonction C est transmis (domaine dépendant du système d'exploitation)

Registre pour valeur renvoyée (virgule flottante)	Registre dans lequel les valeurs à virgule flottante d'appel de fonction C sont renvoyées (domaine dépendant du système d'exploitation)
Ordre d'octet Intel	activé : Structure d'adressage par octet d'Intel est utilisée

9.15.5 Système cible MIPS III ISA, Catégorie Plate-forme de la cible

Boîte de dialogue Paramètres du cible 'MIPS III ISA', plate-forme de la cible



Point du dialogue	**Signification**
Plate-forme	Type de système cible
Premier registre de paramètres (Integer)	Registre dans lequel le premier paramètre (nombre entier) d'appel de fonction C est transmis (domaine dépendant du système d'exploitation)
Dernier registre de paramètre (Integer)	Registre dans lequel le dernier paramètre (nombre entier) d'appel de fonction C est transmis (domaine dépendant du système d'exploitation)
Registre pour valeur renvoyée (Integer)	Registre dans lequel les valeurs entières d'appel de fonction C sont renvoyées (domaine dépendant du système d'exploitation)
Taille max. des arguments sur la pile (octets) :	Dépend du système d'exploitation taille maximale des arguments qui peuvent être transmis dans le projet par le biais de la pile (en octets)

9.15.6 Système cible Hitachi SH, Catégorie Plate-forme de la cible

Boîte de dialogue Paramètres du cible 'Hitachi SH', plate-forme de la cible

Paramètres du cible

Configuration: CoDeSys for HitachiSH

Plate-forme de la cible | Composition du mémoire | Général | Fonctions de réseau

Plate-forme: Hitachi SH

Premier registre de paramètres (Integer): R4
Dernier registre de paramètres (Integer): R7
Registre pour valeur de retour (Intege: R0

☑ Processeur virgule flottante

Premier registre de paramètres (flottant): FR4
Dernier registre de paramètres (flottant): FR7
Registre pour valeur de retour (flottant: FR0

☑ Ordre d'octet Intel

Registrer taille de la zone d'enregistrement: 16

Défaut | OK | Annuler

Point du dialogue	Signification
Plate-forme	Type de système cible
Premier registre de paramètre (Entier)	Registre dans lequel le premier paramètre (nombre entier) d'appel de fonction C est transmis (domaine dépendant du système d'exploitation)
Dernier registre de paramètre (Entier)	Registre dans lequel le dernier paramètre (nombre entier) d'appel de fonction C est transmis (domaine dépendant du système d'exploitation)
Registre pour valeur renvoyée (Entier)	Registre dans lequel les valeurs entières d'appel de fonction C sont renvoyées (domaine dépendant du système d'exploitation)
Processeur virgule flottante	activé : Les commandes FPU sont générées pour les opérations à virgule flottante
Premier registre de paramètres (flottant)	Premier registre pour les paramètres à virgule flottante (domaine dépend du système d'exploitation)
Dernier registre de paramètres (flottant)	Dernier registre pour les paramètres à virgule flottante (domaine dépend du système d'exploitation)
Registre pour valeur renvoyée (flottant)	Registre pour les valeurs renvoyées à virgule flottante (domaine dépend du système d'exploitation)
Ordre d'octet Intel	activé : Structure d'adressage par octet d'Intel est utilisée

9.15.7 Système cible 8051 et compatibles, Catégorie Plate-forme de la cible

Boîte de dialogue Paramètres du cible '8051 et compatibles', plate-forme de la cible

Paramètres du cible

Configuration: CoDeSys for 8051

Plate-forme de la cible | Composition du mémoire | Général | Fonctions de réseau

Plate-forme: 8051 compatible

Taille max. de ls pile:(en octets): 20

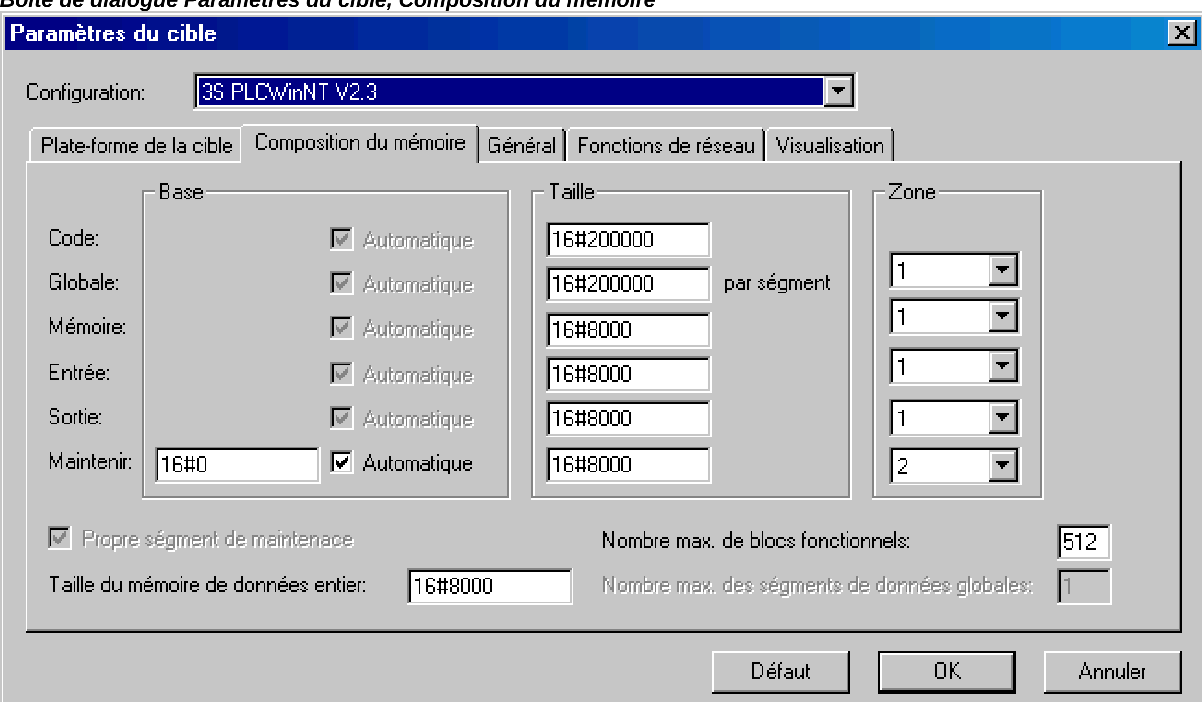
Défaut OK Annuler

Point du dialogue	**Signification**
Plate-forme	Type de système cible
Taille maximale de pile : (en octets)	taille maximale des arguments qui peuvent être transmis par le biais de la pile (nombre d'octets)

9.16 Catégorie: Composition du mémoire

Les points décrits pour cet onglet sont susceptibles d'apparaître pour toutes les plates-formes standard.

Boîte de dialogue Paramètres du cible, Composition du mémoire



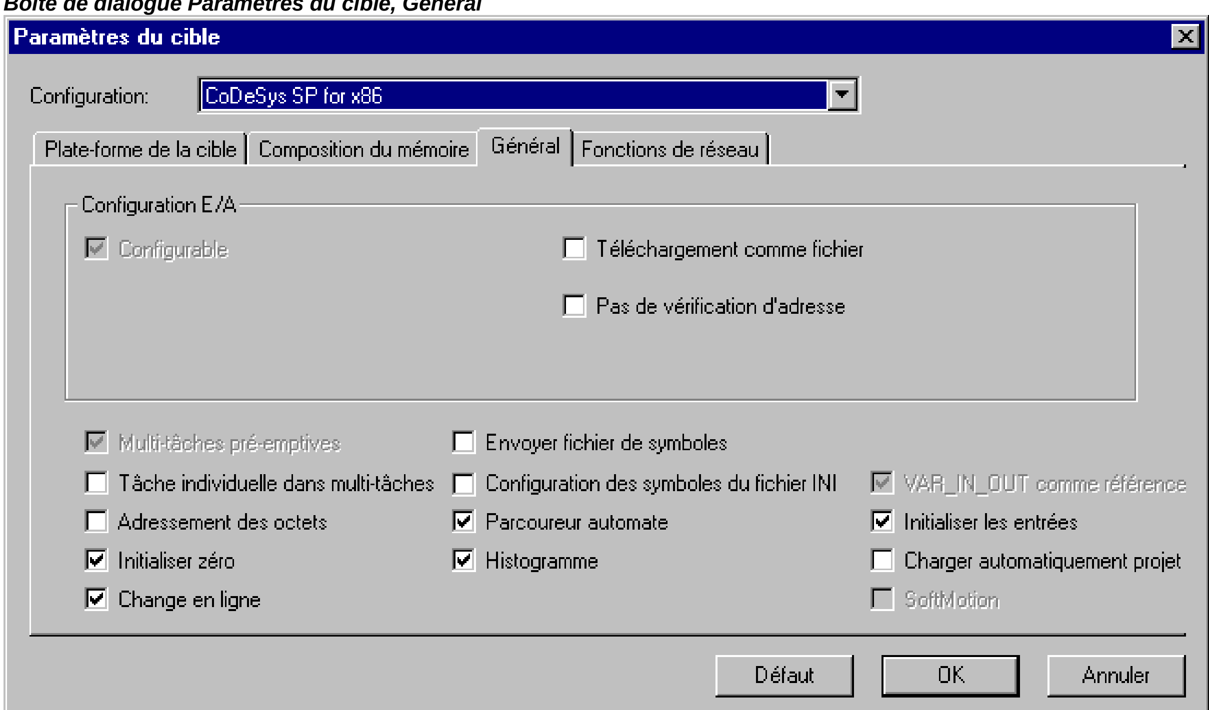
Point du dialogue	Signification
Code (Base)	activé : la zone du code est automatiquement allouée ; désactivé : la zone du code se situe à l'adresse absolue indiquée (Base)
Global (Base)	activé : la zone des données (données globales) est automatiquement allouée dans chaque domaine ; désactivé : la zone des données (données globales) se situe à l'adresse absolue indiquée
Mémoire (Base)	activé : les mémentos sont automatiquement alloués dans chaque domaine ; désactivé : la zone des mémentos se situe à l'adresse absolue indiquée
Entrée (Base)	activé : l'image du processus à l'entrée est automatiquement allouée dans chaque domaine ; désactivé : l'image du processus à l'entrée se situe à l'adresse absolue indiquée
Sortie (Base)	activé : l'image du processus à la sortie est automatiquement allouée dans chaque domaine ; désactivé : l'image du processus à la sortie se situe à l'adresse absolue indiquée
Maintenir (Base)	activé : les données rémanentes sont automatiquement allouées dans chaque domaine ; désactivé : les données rémanentes se situent à l'adresse absolue indiquée
Zone (Code)	Numéro du domaine de la zone des données (Code) Valeurs valides : 1, 2, 3, 4, 5, 6

Zone (Global)	Numéro du domaine de la zone des données (Données globales) Valeurs valides : 1, 2, 3, 4, 5, 6
Zone (Mémoire)	Numéro du domaine de la zone des mémentos Valeurs valides : 1, 2, 3, 4, 5, 6
Zone (Entrée)	Numéro du domaine de l'image du processus à l'entrée Valeurs valides : 1, 2, 3, 4, 5, 6
Zone (Sortie)	Numéro du domaine de l'image du processus à la sortie Valeurs valides : 1, 2, 3, 4, 5, 6
Zone (Maintenir)	Numéro du domaine des données rémanentes Valeurs valides : 1, 2, 3, 4, 5, 6
Base (Code)	Adresse du segment de code (uniquement valide si 'Automatique' n'est pas désactivé)
Base (Global)	Adresse de la zone des données (Données globales) ; (uniquement valide si 'Automatique' n'est pas activé)
Base (Mémoire)	Adresse de la zone des mémentos (uniquement valide si 'Automatique' n'est pas activé)
Base (Entrée)	Adresse de l'image du processus à l'entrée (uniquement valide si 'Automatique' n'est pas activé)
Base (Sortie)	Adresse de l'image du processus à la sortie (uniquement valide si 'Automatique' n'est pas activé)
Base (Maintenir)	Adresse de la zone des données rémanentes (uniquement valide si 'Automatique' n'est pas activé)
Taille (Code)	Taille de la zone du code
Taille par segment (Global)	Taille d'un segment de données
Taille (Mémoire)	Taille de la zone des mémentos
Taille (Entrée)	Taille de l'image du processus à l'entrée
Taille (Sortie)	Taille de l'image du processus à la sortie
Taille (Maintenir)	Taille de la zone des données rémanentes
Propre ségment de maintenance:	activé : les données rémanentes sont gérées dans leur propre segment
Taille du mémoire de données entier:	Taille de l'espace mémoire total
Nombre max. de segments de données globales:	Nombre maximal de segments de données globales qui peut être configuré dans les options du projet
Nombre max. de blocs fonctionnels:	Nombre maximal de modules dans le projet

9.17 Catégorie: Général

Les entrées décrites ici peuvent être utilisées pour toutes les plates-formes standard.

Boîte de dialogue Paramètres du cible, Général



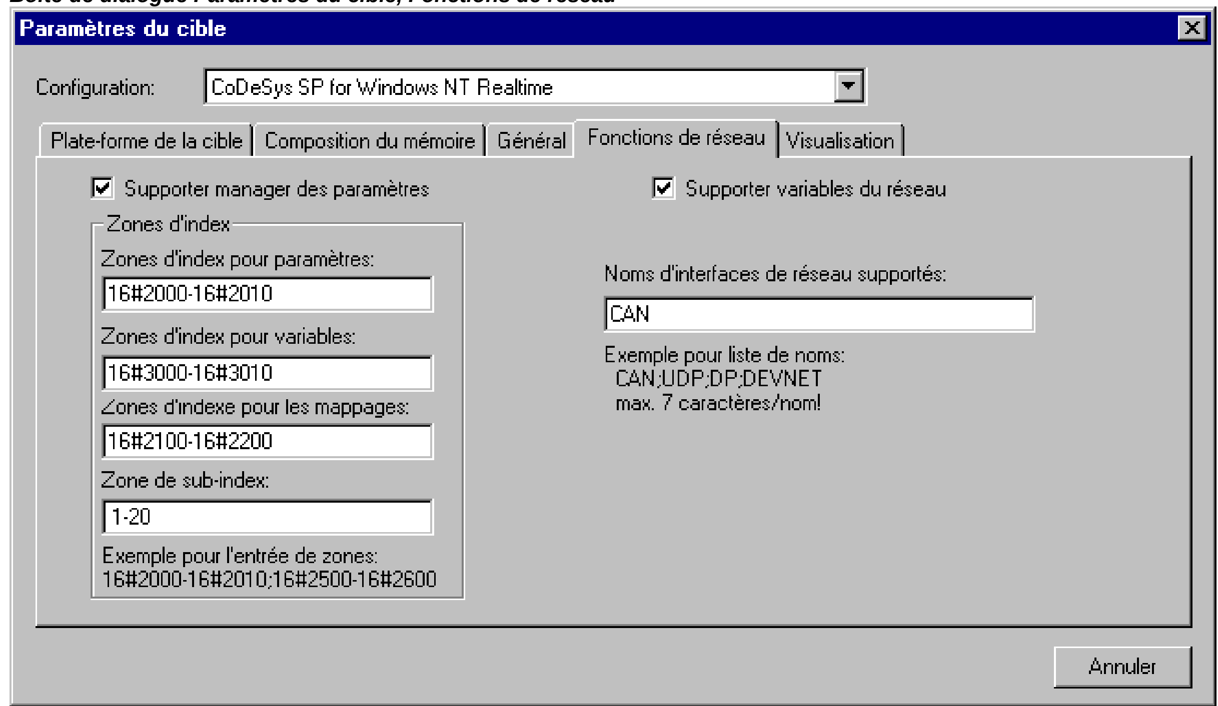
Point du dialogue	Signification
Configurable	activé : supporter une configuration d'E/S configurable et charger la description de la configuration sur l'automate
Supporter configuration CANopen	activé : supporter une configuration CANopen et charger la description de la configuration sur l'automate
Supporter configuration Profibus	activé : supporter une configuration Profibus et charger la description de la configuration sur l'automate
Téléchargement sous forme de fichier	activé : Lors d'un téléchargement, la configuration E/S est chargée sur l'automate sous la forme d'un fichier
Multi-tâches préemptives	activé : supporter la configuration des tâches et charger la description des tâches sur l'automate
Pas de vérification d'adresse	activé : lors de la compilation du projet, les adresses CEI ne sont pas vérifiées
Online Change	activé : Fonctionnalité Online Change (changement en ligne)
Tâche individuelle dans multi- tâches	Ne pas encore implementé
Adressage par octets	activé : l'adressage s'effectue par octets (p.ex. var1 AT %QD4 obtient l'adresse de départ %QB4)
Initialiser zéro	activé : initialisation générale à zéro
Envoyer fichier de symboles	activé : si un fichier de symboles est créé lors d'un téléchargement, il est chargé sur l'automate
Configuration des	activé : Les paramètres pour la configuration des symboles ne sont pas lus du dialogue des options de projet, mais bien du fichier codesys.ini

symboles du fichier INI	ou d'un autre fichier ini s'ils y sont indiqués.
PLC-Browser	activé : fonctionnalité de PLC-Browser
Histogramme	activé : Enregistrement de l'histogramme
VAR_IN_OUT comme référence (à partir de SP4 de V2.2)	activé : Les VAR_IN_OUT sont transmis comme références lors d'un appel de fonction (pointeur); une affectation de constantes ou un accès en lecture / écriture n'est pas possible de l'extérieur
Initialiser les entrées	pas activé : Pour des raisons d'optimisation, aucun code d'initialisation n'est créé pour les entrées déclarées avec "AT %IX" (-> valeurs indéfinies jusqu'au premier cycle de bus!)
Charger automatiquement projet	activé : Après un téléchargement, un projet d'initialisation est automatiquement créé à partir du nouveau programme et envoyé à l'automate.
Softmotion	activé : La fonction SoftMotion est activée, c'est-à-dire disponible sous l'onglet Ressources (Liste de programmes CNC, CAMs).

9.18 Catégorie: Fonctions de réseau

Les entrées décrites ici peuvent être utilisées pour toutes les plates-formes standard.

Boîte de dialogue Paramètres du cible, Fonctions de réseau



Point du dialogue	Signification
Supporter manager des paramètres	activé : l'entrée 'Manager des paramètres' apparaît sous l'onglet Ressources. Celle-ci permet la création des listes pour les variables et les paramètres, ce répertoire servant à un échange de données actif et ciblé avec d'autres automates
Supporter variables du réseau	activé : les variables du réseau peuvent être utilisées afin de servir à l'échange automatique de données au sein du réseau (voir Ressources, Variables réseau)
Noms d'interfaces réseau supportés	Liste des types de réseaux supportés (p.ex. CAN; UDP;)

Zones d'index pour les paramètres	Zone d'index pour les paramètres (voir Ressources, Manager des paramètres)
Zones d'index pour les variables	Zone d'index pour les variables (voir Ressources, Manager des paramètres)
Zones d'index pour les mappings	Zone d'index pour Mappings (voir Ressources, Manager des paramètres)
Zone de sub-index	Zone d'index pouvant être utilisée pour les sub-index au sein des zones d'index ci-dessus pour paramètres ou variables

9.19 Catégorie: Visualisation

Les entrées décrites ici peuvent être utilisées pour toutes les plates-formes standard.

Boîte de dialogue Paramètres du cible, Visualisation

Paramètres du cible

Configuration: 3S CoDeSys SP PLCWinNT V2.4

Plate-forme de la cible | Composition du mémoire | Général | Fonctions de réseau | Visualisation

Largeur de l'affichage en pixe 2000

Hauteur de l'affichage en 2000

Fonts supportés dans le système cible:

☐ Utiliser format de fichier 8.3

☐ Traitement d'alarme dans l'automate

☐ Enregistrement des données de tendance dans l'auto

☐ Activer la variable du système 'CurrentVisu'

☐ Visualisation sur le web

☑ Visualisation sur la cible

☑ Utiliser VISU_INPUT_TASK

Annuler

Point du dialogue	Signification
Largeur d'affichage en pixel **Hauteur d'affichage en pixel**	Une limitation optique selon les valeurs indiquées ici sera affichée lors de la création d'une visualisation dans la fenêtre d'éditeur, afin par exemple de marquer la surface de l'écran sur lequel la visualisation devra ultérieurement être affichée.
Utiliser format de fichier 8.3	Les noms des fichiers bitmap et de langue utilisés dans la visualisation sont automatiquement réduits au format de notation 8.3 et transmis sous ce format au système cible.
Traitement d'alarme dans l'automate	La tâche de Traitement de l'alarme est automatiquement créée dans la Configuration des tâches. Elle exécute de manière implicite un code ST qui évalue l'état de chaque alarme et accomplit le cas échéant les actions s'y rapportant. Pour ce code ST, on a besoin des fonctions d'aide de la bibliothèque SysLibAlarmTrend.lib, et cette dernière est automatiquement chargée. (En outre, les bibliothèques SysLibSockets.lib, SysLibMem.lib,SysLibTime.lib, SysLibFile.lib dont on a également et implicitement besoin sont chargées. Celles-ci doivent être supportées par le système cible !)

Si cette option n'est pas activée alors que la Visualisation Web et/ou la Visualisation Cible le sont, un avertissement sera donné lors de l'ouverture de session.

Remarque: 'Traitement d'alarme dans l'automate' peut également être utilisée lorsque aucune Visualisation Web ou Visualisation Cible n'est activée (voir plus bas). Un code ST est également créé de manière implicite, lequel reprend l'évaluation de l'alarme.

Enregistrement des données de tendance dans l'automate

La tâche TREND_TASK est créée dans la Configuration des tâches. Celle-ci exécute un code ST créé implicitement, lequel enregistre les données de tendance dans une mémoire en anneau, et elle sauvegarde en plus les valeurs dans un système de fichiers, si toutefois l'option Historique est définie au sein de la tendance. Pour ce code ST, on a besoin des fonctions d'aide de la bibliothèque SysLibAlarmTrend.lib, et cette dernière est automatiquement chargée. (En outre, les bibliothèques SysLibSockets.lib, SysLibMem.lib,SysLibTime.lib, SysLibFile.lib dont on a également et implicitement besoin sont chargées. Celles-ci doivent être supportées par le système cible !)

Si cette option n'est pas activée alors que la Visualisation Web et/ou la Visualisation Cible le sont, un avertissement sera donné lors de l'ouverture de session.

Remarque: 'Enregistrement des données de tendance dans l'automate' peut également être utilisée lorsque aucune Visualisation Web ou Visualisation Cible n'est activée (voir plus bas). Un code ST est également créé de manière implicite, lequel reprend l'enregistrement des données de tendance.

Activer la variable du système 'CurrentVisu'

Lorsque cette option est activée, la variable de système CurrentVisu peut être utilisée pour la permutation de visualisations.

Fonts supportes dans le système cible

Liste des polices supportées par le système cible

Visualisation sur le Web

activé : Tous les objets de visualisation du projet sont compilés pour utilisation en tant que visualisations Web. (Chaque objet de visualisation peut cependant en être explicitement exclu dans la boîte de dialogue des Caractéristiques des objets)

Visualisation sur la cible

activé : Tous les objets de visualisation du projet sont compilés pour utilisation en tant que Visualisations Cible. (Chaque objet de visualisation peut cependant en être explicitement exclu dans la boîte de dialogue des Caractéristiques des objets)

Utiliser VISU_INPUT_TASK

(ne peut être activée que si Visualisation cible l'est) Si cette option est activée, deux tâches sont automatiquement créées pour la Visualisation cible (VISU_INPUT_TASK, VISU_TASK); sinon, seul VISU_TASK sera créé, comprenant également les tâches de VISU_INPUT_TASK

Appendice H Utilisation du clavier

Dans le cas où vous pouvez travailler sur CoDeSys uniquement à partir du clavier, vous devez utiliser certaines commandes que vous ne trouvez pas dans le menu:

- La touche de fonction **<F6>** vous permet de basculer de la partie de déclaration à la partie instructions et vice versa, à l'intérieur d'un module ouvert. Dans le Gestionnaire des paramètres, cela vous permet de passer de la fenêtre de navigation à l'éditeur de listes, et vice-versa.
- La combinaison **<Alt>+<F6>** vous permet de basculer d'un objet ouvert vers l'Organisateur d'objets et, de là, vers la fenêtre de messages, si celle-ci est ouverte. Si une boîte de dialogue Rechercher est ouverte, alors la même combinaison vous permet de basculer de l'Organisateur d'objets vers la boîte de dialogue Rechercher et, de là, vers l'objet.
- La touche **<tabulation>** vous permet de vous déplacer d'un champ à l'autre, d'un bouton à l'autre et d'un champ à un bouton, à l'intérieur des boîtes de dialogue.
- Les **touches directionnelles** permettent de se déplacer d'un onglet à l'autre, d'un objet à l'autre et d'un onglet un objet, à l'intérieur de l'Organisateur d'objets et du gestionnaire de bibliothèques.

Toutes les autres actions peuvent être déclenchées à partir des options de menu ou en utilisant les raccourcis indiqués à la suite de ces options. La combinaison **<Maj>+<F10>** vous donne accès au menu contextuel, qui contient les commandes les plus fréquemment utilisées en lien avec un objet sélectionné ou un éditeur actif.

9.20 Combinaisons de touches

Voici un aperçu des combinaisons de touches et des touches de fonction:

Utilisation habituelle

Basculer de la partie de déclaration vers la partie d'instructions d'un module et vice versa	<F6>
Basculer de l'Organisateur d'objets vers l'objet puis vers la fenêtre de messages et vice versa	<Alt>+<F6>
Menu contextuel	<Maj>+<F10>
Passage à la fenêtre d'éditeur suivante et ouverte	<Ctrl>+<F6>
Passage à la fenêtre d'éditeur précédente et ouverte	<Ctrl>+<Maj>+<F6>
Raccourci pour les déclarations	<Ctrl>+<Entrée>
Basculer d'un message dans la fenêtre de messages vers l'endroit correspondant dans l'éditeur	<Entrée>
Afficher et masquer des variables à plusieurs éléments	<Entrée>
Ouvrir et fermer des dossiers	<Entrée>
Déplacement d'un onglet à l'autre dans l'Organisateur d'objets et dans le gestionnaire de bibliothèques	<Touches directionnelles>
Déplacement dans les boîtes de dialogue	<Tabulation>
Aide dépendant du contexte	<F1>

Commandes usuelles

'Fichier' 'Enregistrer'	<Ctrl>+<S>
'Fichier' 'Imprimer'	<Ctrl>+<P>
'Fichier' 'Quitter'	<Alt>+<F4>
'Projet' 'Supprimer un objet'	<Suppr>
'Projet' 'Insérer objet'	<Inser>
'Projet' 'Renommer objet'	<Barre d'espacement>
'Projet' 'Editer objet'	<Entrée>
'Editer' 'Annuler'	<Ctrl>+<Z>
'Editer' 'Refaire'	<Ctrl>+<Y>
'Editer' 'Couper'	<Ctrl>+<X> ou <Maj>+<Suppr>
'Editer' 'Copier'	<Ctrl>+<C>
'Editer' 'Coller'	<Ctrl>+<V>
'Editer' 'Supprimer'	<Suppr>
'Editer' 'Rechercher le suivant'	<F3>
'Editer' 'Liste de sélection pour l'édition'	<F2>
'Editer' 'Erreur prochaine'	<F4>
'Editer' 'Erreur précédente'	<Maj>+<F4>
'En Ligne' 'Démarrer'	<F5>
'En Ligne' 'Point d'arrêt actif/inactif'	<F9>
'En Ligne' 'Etape individuelle sur'	<F10>
'En Ligne' 'Etape individuelle dans'	<F8>
'En Ligne' 'Cycle indépendant'	<Ctrl>+<F5>
'En Ligne' 'Ecrire valeurs des variables'	<Ctrl>+<F7>
'En Ligne' 'Forcer valeurs des variables'	<F7>
'En Ligne' 'Arrêter de forcer'	<Maj>+<F7>
'Fenêtre' 'Messages'	<Maj>+<Echap>

Commandes de l'éditeur du langage FBD

'Insérer' 'Réseau (derrière)'	<Maj>+<T>
'Insérer' - 'Affectation'	<Ctrl>+<A>
'Insérer' 'Saut'	<Ctrl>+<L>
'Insérer' 'Return'	<Ctrl>+<R>
'Insérer' 'Opérateur'	<Ctrl>+<O>
'Insérer' 'Fonction'	<Ctrl>+<F>
'Insérer' 'Bloc fonctionnel'	<Ctrl>+<B>
'Insérer' 'Entrée'	<Ctrl>+<U>

'Extras' 'Négation'	<Ctrl>+<N>
'Extras' 'Zoom'	<Alt>+<Entrée>

Commandes de l'éditeur du langage LD

'Insérer' 'Réseau (derrière)'	<Maj>+<T>
'Insérer' 'Contact'	<Ctrl>+<O>
'Insérer' 'Contact parallèle'	<Ctrl>+<R>
'Insérer' 'Bloc fonctionnel'	<Ctrl>+<B>
'Insérer' 'Bobinage'	<Ctrl>+<L>
'Extras' 'Insérer ci-dessous'	<Ctrl>+<U>
'Extras' 'Négation'	<Ctrl>+<N>

Commandes de l'éditeur SFC

'Insérer' 'Etape-Transition (devant)'	<Ctrl>+<T>
'Insérer' 'Etape-Transition (derrière)'	<Ctrl>+<E>
'Insérer' 'Séquence alternative (droite)'	<Ctrl>+<A>
'Insérer' 'Séquence simultanée (droite)'	<Ctrl>+<L>
'Insérer' 'Saut' (SFC)	<Ctrl>+<U>
'Extras' 'Zoom action/transition'	<Alt>+<Entrée>
Basculer à nouveau vers l'éditeur à partir de l'aperçu du diagramme fonctionnel en séquence	<Entrée>

Utilisation de la configuration de l'automate ou de la configuration des tâches

Ouvrir et fermer des éléments d'organisation	<Entrée>
Mettre un cadre d'édition autour du nom	<Barre d'espacement>

Utilisation de l'Éditeur du gestionnaire des paramètres

Passer de la fenêtre de navigation à l'éditeur de listes et vice-versa	<F6>
Effacement d'une ligne dans l'éditeur de listes	<Ctrl>+<Suppr>, <Maj>+<Suppr>
Effacement d'un champ	<Suppr>

Appendice I Messages d'erreur et Avertissements

Lors de la compilation du projet, les messages relatifs aux erreurs éventuellement survenues et aux avertissements sont affichés dans la fenêtre de messages. <F4> permet de sauter à la ligne de messages suivante, ce qui provoque l'ouverture de la fenêtre à l'endroit concerné du programme. Dans la fenêtre de messages, les messages d'erreur et les avertissements sont précédés d'un numéro d'ID univoque. Si une telle ligne de message est marquée, <F1> permet d'ouvrir la fenêtre d'aide correspondante.

9.20.1 Avertissements

1100

"Service est inconnu '<nom>' dans la bibliothèque."
Vous utilisez une bibliothèque externe. Vérifiez si toutes les fonctions données dans le fichier en format hexadécimal (.hex) sont également définies dans le fichier .lib.

1101

"Symbole pas résolu '<Symbol>'."
Le générateur de codes attend un module du nom <Symbol>. Celui-ci n'est pas défini dans le projet. Veuillez définir une fonction/un programme avec le nom adéquat.

1102

"Interface invalide pour symbole '<Symbol>'."
Le générateur de codes attend une fonction du nom <Symbol> avec précisément une entrée scalaire ou un programme du nom <Symbol> sans entrée ou sortie.

1200

"Tâche '<nom>', appel à '<nom>': Les variables d'accès dans la liste de paramètres n'ont pas été mises à jour."
Les variables qui ne sont utilisées que lors d'un seul appel de bloc fonctionnel dans la configuration des tâches ne sont pas reprises dans la liste des références croisées.

1300

"Le fichier '<nom>' n'a pas été trouvé"
Le fichier auquel renvoie l'objet de variable globale n'existe pas. Vérifiez le chemin d'accès.

1301

"La bibliothèque d'analyse n'a pas été trouvée! Le code pour l'analyse n'est pas généré."
Vous utilisez la fonction d'analyse, mais la bibliothèque analyzation.lib n'est pas disponible.
Insérez cette bibliothèque dans le gestionnaire des bibliothèques.

1400

"La directive du compilateur '<nom>' est ignorée!"
Cette pragma n'est pas supportée par le compilateur. Reportez-vous au mot-clé 'Pragma' pour les directives supportées.

1401

"La structure '<nom>' ne contient pas d'éléments."
La structure ne contient aucun élément, mais les variables de ce type occupent un octet dans la mémoire.

1500

"Cette expression ne contient pas d´assignement. Aucun code n'est généré."
Le résultat de cette expression n'est pas utilisé. Ainsi, aucun code n'est généré pour la totalité de l'expression.

1502

"La variable '<nom>' porte le même nom qu'un module. Le module ne sera pas appelé!"
Vous utilisez une variable qui porte le même nom qu'un module.
Exemple :

```
PROGRAM a
...
VAR_GLOBAL
        a: INT;
END_VAR
...
a; (* Le module a n'est pas appelé, mais la variable a est chargée. *)
```

1501

"La constrante de chaîne est passée comme 'VAR_IN_OUT': '<nom>' ne doit pas être écrasé!"
On ne peut écrire la constante dans le corps du module puisque aucune vérification de taille n'y a lieu.

1503

"Le module '<nom>' n'a pas de sorties. Le résultat est mis à 'TRUE'."
Vous continuez de lier le pin de sortie d'un module sans sorties en langage FBD ou LD. La liaison se voit automatiquement attribuer la valeur TRUE.

1504

"L'instruction ne sera probablement pas exécutée à cause de l'expression logique."
Toutes les branches de l'expression logique ne sont pas éventuellement exécutées.
Exemple :
IF a AND funct(TRUE) THEN ....
Si a a la valeur FALSE, funct ne sera plus appelé.

1505

"Effet latéral dans '<nom>'! Probablement, la branche n'est pas calculée!"
La première entrée du module a la valeur FALSE, il se peut dès lors que la branche latérale débouchant à la seconde entrée ne soit plus évaluée.

1600

"Pas d'évidence quel bloc de données vous voulez ouvrir (le code généré contient éventuellement des erreurs)."
Le programme original Siemens ne permet pas de déterminer quel module de données est ouvert.

1700

"L'entrée n'est pas liée."
Vous utilisez une boîte d'entrée en CFC qui n'a pas de liens consécutifs. Aucun code n'est alors créé.

1800

"<nom>(Elemente #<Nombre elemente>): Expression d'espion invalide '<nom>'"
L'élément de visualisation contient une expression qui ne peut être espionnée. Vérifiez le nom de variable et le remplacement d'espaces réservés.

1801

"Entrée sur l'expression '<nom>' pas possible."
Dans la configuration de l'objet de visualisation, vous utilisez une expression composée comme cible d'une action d'entrée. Remplacez cette expression par une variable individuelle.

1900

"L'unité de programmation '<nom>' (routine principale) n'est pas disponible dans la bibliothèque."
Le module d'entrée (p.ex.PLC_PRG) ne sera pas disponible lors de l'utilisation de la bibliothèque.

1901

"Les variables d'accès et de configuration ne sont pas enregistrées dans la bibliothèque!"
Les variables d’accès et la configuration de variables ne seront pas sauvegardées dans la bibliothèque.

1902

"'<nom>': Bibliothèque n'est pas approprié pour le type de machine ou erronée!"
Le fichier .obj de la bibliothèque a été créé pour un autre type de machine.

1903

"<nom>: n'est pas de bibliothèque valide."
Le fichier ne correspond pas au format requis de fichier du système cible

9.20.2 Erreurs

3100

"Programme trop grand. Volume max: '<Nombre>' Byte (<Nombre>K)"
La taille maximale du programme est dépassée. Veuillez réduire le programme.

3101

"Volume de données trop grand. Volume max: '<Nombre>' Byte (<Nombre>K)"
L’espace mémoire est complètement utilisé. Réduisez les besoins en fichiers de votre application.

3110

"Erreur dans le fichier de la bibliothèque '<nom>'."
Le fichier .hex ne correspond pas au format INTEL Hex.

3111

"Bibliotheque '<nom>' trop grand. Volume max: 64K"
Le fichier .hex dépasse la taille maximale admissible.

3112

"Instruction ne peut pas déplacer dans la bibliothèque."
Le fichier .hex contient une instruction non relocalisable. Le code de bibliothèque ne peut être lié.

3114

"La bibliothèque utilise plus qu'un ségment."
Les tableaux et le code compris dans le fichier .hex utilisent plus qu’un segment.

3113

"Le code de la bibliothèque écrase les tableaux de fonctions."
Les domaines du code et des tableaux d’information se recoupent.

3115

"La constante ne peut pas être assignée à VAR_IN_OUT. Types de données incompatibles."
Le format interne de pointeur pour les constantes de chaînes de caractères ne peut être converti au format interne de VAR_IN_OUT, car les données sont définies comme "near" et les constantes de chaînes comme "huge" ou "far". Modifiez si possible cette configuration du système cible.

3121

"L'unité de programmation est trop grand."
Un module ne peut dépasser la taille de 64K.

3122

"Initialisation trop grande. Dim. max.: 64K"
Le code d’initialisation pour un bloc fonctionnel ou une structure ne peut dépasser la taille de 64K.

3120

"Le ségment de code actuel excède 64K."
Le code système généré à l’instant est plus grand que 64K. Il se peut qu’il y ait trop de code d’initialisation requis.

3130

" Stack d'application trop petit: <Anzahl> 'DWORD' nécessaire, <Anzahl> 'DWORD' disponible."
Le niveau d’imbrication de l’appel de module est trop grand. Augmentez la taille de pile dans la configuration du système cible ou compilez le programme sans utiliser l’option de compilation de projet 'Débogage'.

3131

" Stack d'utilisateur trop petit: <Nombre> 'DWORD' nécessaire, <Nombre> 'DWORD' disponible."
Veuillez vous adresser au fabricant de l’automate.

3132

" Stack du systeme trop petit: <Nombre> 'DWORD' nécessaire, <Nombre> 'DWORD' disponible."
Veuillez vous adresser au fabricant de l’automate.

3150

"Paramètre '<nom>' de la fonction '<nom>': Le résultat d'une fonction CEI ne peut pas être passé comme paramètre de chaîne d'une fonction C."
Utilisez une variable intermédiaire qui recevra le résultat de la fonction CEI.

3161

"La bibliothèque '<nom>' ne contient pas de ségment de code."
Un fichier .obj de bibliothèque doit contenir au moins une fonction C. Insérez dans le fichier .obj une fonction fictive qui n’est pas définie dans le fichier .lib.

3160

"Ne peut pas ouvrir '<nom>'."
Le fichier '<Nom>' requis pour une bibliothèque ne peut être trouvé.

3163

"Type de référence inconnu dans la bibliothèque '<nom>' (Symbol '<nom>' , Class '<nom>' , Type '<nom>')"
Le fichier .obj contient un type de référence que le générateur de code ne peut résoudre. Vérifiez la configuration du compilateur C.

3200

"<nom>: Expression logique trop complexe."
L’espace mémoire temporaire du système cible n’est pas suffisant pour la taille de l’expression. Divisez l’expression en plusieurs expressions partielles avec attribution à des variables intermédiaires.

3162

"Ne peut pas résoudre la référence dans la bibliothèque (Symbole '<nom>' , Class '<nom>' , Type '<nom>')"
Le fichier .obj contient une référence non résoluble à un autre symbole. Vérifiez la configuration du compilateur C.

3201

"<nom>: Un réseau ne doit produire plus que 512 bytes de code"
Les sauts internes ne peuvent être résolus. Activez l’option "Utiliser des décalages de saut 16 bit" dans la configuration cible 68K.

3202

"Débordement de la pile en cas d'appels de fonction cordon/zone/structure trop complexes"
Vous utilisez un appel de fonction imbriqué de la forme CONCAT(x, f(i)). Ceci peut provoquer une perte de données. Divisez l’appel en deux expressions.

3203

"Seulement deux variables de système par affectation"
Divisez l'affectation en plusieurs parties.

3204

"Un saut excède 32k octets."
La distance d'un saut ne peut être supérieure à 32767 octets.

3205

"Erreur interne: Trop de chaînes constantes"
Un maximum de 3000 constantes de chaînes de caractères peut être utilisé dans un module.

3206

"Le bloc fonctionnel est trop grand."
Un bloc fonctionnel se compose d'un maximum de 32767 octets de code.

3207

"Optimation du tableau."
L'optimisation de l'accès au tableau a échoué car une action a été appelée au sein du calcul de l'index.

3208

"La conversion n'est pas encore implémentée."
Vous utilisez une fonction de conversion qui n'est pas implémentée pour le générateur de code actuel.

3209

"Opérateur pas implémenté."
Vous utilisez un opérateur qui n'est pas implémenté pour le générateur de code actuel pour ce type de données : MIN(string1,string2).

3210

"Fonction '<nom>' pas trouvée."
Vous appelez une fonction qui n'est pas disponible au sein du projet.

3211

"Usage maximum de la variable de chaîne excédé."
Une variable de type STRING ne peut être utilisée que dix fois dans une expression et dans le cas d'un générateur de code 68K.

3250

"Real n'est pas accepté sur les automates 8 bits."
Le système cible n'est pas supporté pour l'instant.

3251

"Types 'date du jour' ne sont pas acceptés pour les automates 8 bits."
Le système cible n'est pas supporté pour l'instant.

3252

"Taille du stack excède <number> Bytes"
Le système cible n'est pas supporté pour l'instant.

3253

"Ne pouvait pas trouver le fichier hex: '<nom>' "
Le système cible n'est pas supporté pour l'instant.

3254

"L'appel d'une fonction de bibliothèque externe ne pouvait être résolu."
Le système cible n'est pas supporté pour l'instant.

3400

"Erreur à l'importation des variables d'accès"
Le fichier .exp contient une rubrique incorrecte de variables d’accès.

3401

"Erreur à l'importation de la configuration des variables"
Le fichier .exp contient une rubrique incorrecte de variables de configuration.

3402

"Erreur à l'importation des variables globales"
Le fichier .exp contient une rubrique incorrecte de variables globales.

3403

"<nom> ne pouvait être importé"
La rubrique relative à l’objet donné dans le fichier .exp est incorrecte.

3404

"Erreur à l'importation de la configuration des tâches"
La rubrique relative à la configuration de tâche dans le fichier .exp est incorrecte.

3405

"Erreur à l'importation de la configuration de l'automate"
La rubrique relative à la configuration de l’automate dans le fichier .exp est incorrecte.

3406

"Deux pas nommés '<nom>'! Le deuxième pas n'a pas été importé"
La rubrique du module SFC dans le fichier .exp comprend deux étapes portant le même nom. Renommez une de ces deux étapes dans le fichier d’exportation.

3407

"Pas d'éntrée '<nom>' pas trouvé"
L’étape nommée n’existe pas dans le fichier .exp.

3408

"Pas suivant '<nom>' pas trouvé"
L’étape nommée n’existe pas dans le fichier .exp.

3409

"Pas de transition suivante pour le pas '<nom>' "
La transition requise par l’étape indiquée comme étape d’entrée, n’existe pas dans le fichier .exp.

3410

"Pas de pas suivant pour la transition '<nom>' "
La transition indiquée, requise par l’étape, n’existe pas dans le fichier .exp.

3411

"Le pas '<nom>' n'est pas accessible du pas init"
Le lien entre l’étape indiquée et l’étape d’initialisation n’existe pas dans le fichier .exp.

3450

"PDO<nom>: <nom du module> <nom du dialogue configuration>-<nom PDO> COB-Id manque!"
Cliquez sur le bouton Caractéristiques dans la boîte de dialogue de configuration <nom de la boîte de dialogue de configuration> du module <nom du module>, et entrez une COB ID pour le PDO <nom de PDO>.

3451

"Erreur en chargeant: Le fichier EDS '<nom>' n'a pas été trouvé, mais est utilisé dans la configuration!"

Le fichier d’appareil nécessité par la configuration CAN n’est probablement pas dans le répertoire correct. Vérifiez ce point par le biais de l’entrée de répertoire pour les fichiers de configuration dans 'Projet' 'Options' 'Répertoires'.

3452

"Le module '<nom>' ne pouvait être crée!"
Le fichier d’appareil pour le module <Nom> ne correspond pas à la configuration présente. Il a été probablement modifié depuis la création de la configuration ou le fichier est corrompu.

3453

"Le canal '<nom>' ne pouvait être crée!"
Le fichier d’appareil pour le canal <Nom> ne correspond pas à la configuration présente. Il a été probablement modifié depuis la création de la configuration ou le fichier est corrompu.

3454

"L'adresse '<adresse>' renvoie à une plage de mémoire occupée!"
Vous avez activé l’option 'Vérifier superposition d’adresses' dans la boîte de dialogue de la configuration de l’automate, et une superposition a été constatée. Veuillez noter que la base de la vérification est la taille résultant du type de données et non la valeur de 'size' dans le fichier de configuration.

3455

"Erreur en chargeant: Le fichier GSD '<nom>' n'a pas été trouvé, mais est utilisé dans la configuration!"
Le fichier d’appareil nécessité par la configuration Profibus n’est probablement pas dans le répertoire correct. Reportez-vous à l’entrée pour les fichiers de configuration dans 'Projet' 'Options' 'Répertoires'.

3456

"L'appareil Profibus '<nom>' n'a pas pu être crée!"
Le fichier d'appareil pour l'appareil <nom> ne correspond pas à la configuration présente. Il a été probablement modifié depuis la création de la configuration ou le fichier est corrompu.

3457

"Erreur dans la description du module!"
Vérifiez le fichier d'appareil appartenant au module.

3500

"Pas de VAR_CONFIG pour '<nom>' "
Insérez une déclaration dans la liste des variables globales (contenant la 'Configuration de variables') pour la variable nommée.

3501

"Pas d'adresse dans VAR_CONFIG pour '<nom>' "
Insérez une adresse dans la liste des variables globales (contenant la 'Configuration de variables') pour la variable nommée.

3502

"Faux type de données de '<nom>' dans VAR_CONFIG"
La variable nommée est déclarée avec un autre type de données dans la liste des variables globales contenant la configuration des variables que dans le bloc fonctionnel.

3503

"Faux type d'adresse de '<nom>' dans VAR_CONFIG"
La variable nommée est déclarée avec un autre type d'adresse dans la liste des variables globales contenant la configuration des variables que dans le bloc fonctionnel.

3504

" Valeurs initiales pour les variables 'VAR_CONFIG' ne sont pas supportées"
Une variable de la configuration des variables est déclarée avec adresse et valeur initiale. Une valeur initiale ne peut cependant être définie que dans le cas d'une variable d'entrée sans attribution d’adresse.

3505

"<nom> n'est pas de chemin d'instance valable "
Une variable n’existant pas est indiquée dans la configuration des variables.

3506

"Chemin d´accés est attendu "
Dans la liste globale des variables pour les variables d’accès, il manque le chemin correct d’accès pour une variable :
<Identificateur> **:** ‘<Chemin d’accès>’ **:** <Type> <Type d’accès>

3507

"Énoncé de l`adresse n`est pas admis pour 'VAR_ACCESS'"
Dans la liste globale des variables pour les variables d’accès, il existe une affectation d'adresse pour une variable :
Définition correcte : <Identificateur> **:** ‘<Chemin d’accès>’ **:** <Type> <Type d’accès>

3550

"Le nom de tâche '<nom>' a été utilisé deux fois"
Vous avez défini deux tâches avec le même nom. Renommez-en une.

3551

" La tâche '<nom>' doit contenir au moins un appel de programme "
Insérez un appel de programme ou effacez la tâche.

3552

" La variable d'événement '<nom>'dans la tâche '<nom>' n'est pas définie. "
Vous avez utilisé dans la configuration de la tâche nommée une variable événementielle qui n’est pas déclarée globalement dans le projet. Utilisez une autre variable ou définissez la variable en question de manière globale.

3553

" La variable d'événement '<nom>' dans la tâche '<nom>' doit être du type 'BOOL' "
Utilisez une variable de type BOOLéene comme variable événementielle.

3554

" L'entrée de tâche '<nom>' doit être un programme ou une instance de bloc fonctionnel globale. "
Vous avez entré une fonction ou un module non défini dans le champ Appel de programme.

3555

" L'entrée de tâche '<nom>' contient des paramètres faux. "
Vous avez entré dans le champ Appel de programme des paramètres qui ne correspondent pas à la déclaration du module.

3600

"Variable implicite non trouvée!"
Appliquez tout d’abord la commande 'Compiler tout'. Si le message d’erreur apparaît à nouveau, veuillez vous adresser au fabricant de l’automate.

3601

"<nom> est un nom de variable réservé"
Vous avez déclaré une variable au sein du projet qui est déjà réservée pour le générateur de code. Renommez cette variable.

3610

" '<nom>' n'est pas supporté"
La caractéristique donnée n’est pas supportée par cette version.

3611

" Le répertoire de compilation '<nom>' est non valide."
Vous avez entré un répertoire incorrect pour les fichiers de compilation dans les options de projet / répertoires.

3612

" Le nombre maximum de modules ('<numéro>') a été excédé! La compilation sera abandonnée."
Vous utilisez trop de modules et de types de données dans le projet. Changez le nombre maximal de modules dans la configuration du système cible / Répartition de la mémoire.

3613

"Compilation abandonnée "
La compilation a été interrompue par l’utilisateur.

3614

" Le projet doit contenir un module avec le nom '<nom>' (routine principale) ou une configuration de tâche. "
Un projet nécessite une entrée au programme du type programme (p.ex. PLC_PRG) ou une configuration de tâches.

3615

"<nom>' (Einsprungfunktion) doit être de type programme "
Vous utilisez une entrée au programme(p.ex. PLC_PRG) qui n’est pas du type programme.

3616

"Les programmes contenant des données ne sont pas acceptés par les bibliothèques externes. "
La bibliothèque à enregistrer contient un programme. Ce dernier ne sera pas à disposition lors de l’utilisation de la bibliothèque.

3617

"Trop peu de mémoire "
Augmentez la mémoire virtuelle de votre calculateur.

3618

" Les accès de bits ne sont pas acceptés du générateur de code actuelle. "
Le générateur de code pour le système cible actuellement réglé ne supporte pas les accès bit à bit aux variables.

3700

" Un module appelé'<nom>' existe déjà dans la bibliothèque '<nom>' "
Vous utilisez un nom de module qui est déjà attribué à un module de bibliothèque. Renommez le module.

3701

"Le nom du module dans le nom de déclaration ne correspond pas au nom dans la liste des objets"
Renommez votre module au moyen de la commande de menu "**Projet**" + "**Renommer objet**" ou modifiez le nom du module au niveau de sa partie de déclaration. Le nom doit suivre immédiatement les mots-clés PROGRAM, FUNCTION ou FUNCTIONBLOCK.

3702

" Trop d'idenfificateurs "
Un maximum de 100 identificateurs peut être attribué pour chaque déclaration de variable.

3703

" Plusieurs déclarations possèdent des identificateurs identiques '<nom>' '"
Plusieurs identificateurs portant le même nom existent dans la partie déclaration de l’objet.

3704

" Récursion de données: <module 0> -> <module 1> -> .. -> <module 0>"
Une instance de bloc fonctionnel se nécessitant elle-même a été utilisée.

3705

"'<nom>': 'VAR_IN_OUT' n'est pas admis dans les modules de haut niveau quand il n'y a pas de configuration des tâches"
Établissez une configuration de tâches ou assurez-vous qu'aucune variable VAR_IN_OUT ne soit utilisée dans PLC_PRG.

3720

" Adresse attendue après 'AT' "
Insérez une adresse valide après le mot clé AT ou changez le mot clé AT.

3721

" Seulement 'VAR' et 'VAR_GLOBAL' peuvent être placés sur des adresses "
Copiez la déclaration dans une zone VAR ou VAR_GLOBAL.

3722

" Sur une adresse de bit, seulement des variables 'BOOL' sont admises "
Changez l'adresse ou le type de variable donné dans la déclaration.

3726

" Une constante ne peut pas être posée sur une adresse directe. "
Modifiez l'affectation d'adresse en conséquence.

3727

"Tableaus ne peuvent pas mettre á cette adresse "
Modifiez l'affectation d'adresse en conséquence.

3728

"Adresse inadmissible: '<Adresse>'"
Cette adresse n'est pas supportée par la configuration de l'automate. Vérifiez la configuration ou corrigez l'adresse.

3729

"Type interdit '<nom>' sur l'adresse: '<nom>'"
Le type de cette variable ne peut être placé à l'adresse indiquée. Exemple : pour un système cible qui travaille avec un alignement de 2 octets (Alignment 2), la déclaration suivante est non valide : var1 AT %IB1:WORD;

3740

" Type inconnu: '<nom>' "
Vous utilisez un type non valide pour la déclaration de variables.

3741

" Identificateur de type attendu "
Vous utilisez un mot clé ou un opérateur à la place d'un identificateur de type valide.

3742

" Valeur d'énumération attendue "
Dans la définition du type d'énumération, il manque un identificateur après la parenthèse ouvrante ou après une virgule dans la zone délimitée par les parenthèses.

3743

"Nombre entier attendue "
Les valeurs énumératives ne peuvent être initialisées qu'avec des nombres entiers du type INT.

3744

"La constante enum '<nom>' est déjà définie."
Vérifiez si vous avez respecté les règles suivantes lors de l'attribution de valeurs énumératives :
- toutes les valeurs doivent être univoques au sein d'une définition d'énumération.
- toutes les valeurs doivent être univoques au sein des définitions globales d'énumération.
- toutes les valeurs doivent être univoques au sein des définitions locales d'énumération d'un module.

3745

"Les limites de plage ne sont admises que pour les types de données intègres!"
Les types domaines partiels ne peuvent être définis que sur base de types de données entières.

3746

"La limite de plage '<nom>' n'est pas compatible au type de données '<nom>'."
Une des limites données pour le type domaine partiel est située en dehors des limites autorisées pour le type de base.

3747

"Longueur de chaîne de caractères inconnue: '<nom>'"
Vous utilisez une constante inconnue pour la définition de la longueur de la chaîne de caractères.

3748

"Un tableau ne doit avoir plus de trois dimensions."
Vous utilisez plus que les trois dimensions autorisées pour un tableau. Utilisez le cas échéant un ARRAY OF ARRAY.

3749

"La limite inférieure '<nom>' est inconnue."
Vous utilisez une constante non définie comme limite inférieure d'un type domaine partiel ou d'un type tableau (ARRAY).

3750

"La limite supérieure '<nom>' est inconnue."
Vous utilisez une constante non définie comme limite supérieure d'un type domaine partiel ou d'un type tableau (ARRAY).

3760

"Valeur d'initiale fausse"
Utilisez une valeur initiale qui corresponde à la définition du type. Vous pouvez vous aider de la boîte de dialogue de déclaration de variables (Maj/F2 ou 'Editer' 'Déclaration de variables').

3761

"Variables 'VAR_IN_OUT' ne doivent pas avoir de valeur initiale."
Enlevez l'initialisation dans la déclaration de variables.

3780

"'VAR', 'VAR_INPUT', 'VAR_OUTPUT' ou 'VAR_IN_OUT' attendu"
La première ligne qui suit le nom d'un module, doit contenir un de ces mots-clés.

3781

"'END_VAR' ou identificateur attendu"
Ecrivez un identificateur valide ou END_VAR au début de la ligne de déclaration.

3782

"Fin inattendue"
Dans la partie de déclaration : insérez le mot-clé END_VAR à la fin de la partie de déclaration.
Dans l'éditeur littéraire : insérez des instructions qui complètent la dernière séquence d'instructions (par exemple END_IF).

3783

"'END_STRUCT' ou identificateur attendu"
Assurez-vous que la déclaration de type soit correctement terminée.

3800

"Les variables globales occupent trop de mémoire. Veuillez augmenter le mémoire disponible dans les options de projet."
Augmentez le nombre de segments réglé pour les options de compilation dans les options de projet.

3801

"La variable '<nom>' est trop grande ('<nombre>' Bytes)"
La variable utilise un type qui est plus grand qu'un segment de donnée. Vous pouvez régler la longueur de segment, selon le système cible, dans la Configuration du système cible / Répartition de la mémoire. Si vous n'y trouvez aucune possibilité de saisie, veuillez vous adresser au fabricant de l'automate.

3802

"Le mémoire pour la variable de retain est épuisé. Variable '<nom>', octets."
L'espace mémoire disponible pour les variables rémanentes est épuisé. Vous pouvez régler cet espace, selon le système cible, dans la Configuration du système cible / Répartition de la mémoire. Si vous n'y trouvez aucune possibilité de saisie, veuillez vous adresser au fabricant de l'automate. (Veuillez également noter à cet effet, que dans le cas d'instances de blocs fonctionnels dans lesquels une variable rémanente est utilisée, l'instance complète est gérée dans la zone de mémoire rémanente !)

3803

"Le mémoire pour les variables globales est épuisé.."
L'espace mémoire disponible pour les variables globales est épuisé. Vous pouvez régler cet espace, selon le système cible, dans la Configuration du système cible / Répartition de la mémoire. Si vous n'y trouvez aucune possibilité de saisie, veuillez vous adresser au fabricant de l'automate.

3820

"'VAR_OUTPUT' et 'VAR_IN_OUT' ne sont pas admis dans les fonctions."
Vous ne pouvez définir aucun paramètre de référence / de sortie dans une fonction.

3821

"Au moins une entrée est nécessaire pour une fonction"
Insérez au moins un paramètre d'entrée pour la fonction.

3840

"Variable globale inconnue '<nom>'!"
Vous utilisez dans le module une variable VAR_EXTERNAL pour laquelle aucune variable globale correspondante n'est déclarée.

3841

"Déclaration de '<nom>' ne correspond pas à la déclaration globale!"
La définition du type dans la déclaration de la variable de type VAR_EXTERNAL ne correspond pas à la définition dans la déclaration globale.

3900

"Soulignes multiples dans l'identificateur"
Enlevez les caractères de soulignement multiples dans l'identificateur.

3901

"Un maximum de 5 zones numériques est admis dans les adresses"
Vous utilisez une affectation directe d'adresse pour une adresse qui contient plus de quatre niveaux (p.ex. %QB0.1.1.0.1).

3902

"Des mots clé s'écrivent en majuscules"
Écrivez le mot clé en majuscules ou activez l'option 'Formater automatique'.

3903

"Constante de temps inadmissible"
La constante n'est pas donnée conformément au format IEC61131-3.

3904

"Constante de temps dépasse valeur maximum"
Vous utilisez une valeur pour la constante de temps qui ne peut plus être représentée en format interne. La valeur maximale représentable est t#49d17h2m47s295ms.

3906

"Constante d'heure du jour inadmissible"
La constante n'est pas donnée conformément au format IEC61131-3.

3907

"Constante date/temps inadmissible"
La constante n'est pas donnée conformément au format IEC61131-3.

3908

"Chaîne de caractères non valid "
La constante de chaîne contient un caractère non valide.

4000

"Identificateur attendu "
Introduisez un identificateur valide à cet endroit.

4001

"Variable '<nom>' non déclarée"
Déclarez la variable locale ou globale.

4010

"Types incompatibles: Ne peut pas convertir '<nom>' en '<nom>'."
Vérifiez les types requis par l'opérateur (recherchez pour cela l'opérateur dans votre fichier d'aide) et remplacez le type de la variable qui a généré l'erreur par un type admis ou choisissez une autre variable.

4011

"Type inadmissible dans paramètre '<nombre>' de '<nom>': Ne peut pas convertir '<nom>' en '<nom>'."
Le type du paramètre actuel ne peut être converti à celui du paramètre formel. Utilisez une conversion de type ou un type de variable correspondant.

4012

"Type inadmissible pour paramètre '<nom>' de '<nom>': Ne peut pas convertir '<nom>' en '<nom>'."
Une valeur du type inadmissible <type2> est affectée à la variable '<nom>'. Remplacez la variable ou la constante par une variable ou une constante de type <type1>, ou utilisez une conversion de type ou une constante avec préfixe de type.

4013

"Type inadmissible pour sorties '<nom>' de '<nom>': Ne peut pas convertir '<nom>' en '<nom>'."
Une valeur du type inadmissible <type2> est affectée à la variable '<nom>'. Remplacez la variable ou la constante par une variable ou une constante de type <type1>, ou utilisez une conversion de type ou une constante avec préfixe de type.

4014

" Constante avec préfixe de type: '<nom>' ne peut pas être converti en '<nom>'. "
Le type de la constante n'est pas compatible avec le type du préfixe.

Exemple : SINT#255

4015

" Type de données non valide <'<nom>'> pour accès de bit direct. "
L'adressage direct de bits n'est autorisé que pour les types de données entières (Integer) et chaînes de bits (Bitstring). Vous utilisez dans l'accès bit à bit <var1>.<bit> une variable var1 de type REAL/LREAL ou une constante.

4016

" L'index de bit '<numéro>' est hors de la plage valide pour variables du type %s'. '"
Vous essayez d'accéder à un bit qui n'est pas défini pour le type de donnée de la variable.

4017

" 'MOD' n'est pas defini pour 'REAL' "
L'opérateur MOD ne peut être utilisé que pour les types de données entières (Integer) et chaînes de bit (Bitstring).

4020

" Les opérandes 'ST', 'STN', 'S', 'R' doivent être des variables modifiables"
Remplacez le premier opérande par une variable à laquelle l'on peut accéder en écriture.

4021

" Pas de permission de modification pour '<nom>' "
Remplacez la variable par une variable avec accès en écriture.

4022

"Opérande attendu"
Ajoutez un opérande derrière la commande existante.

4023

" Après '+' ou '-', un nombre est attendue "
Entrez un chiffre.

4024

"Erwarte <Operator 0> oder <Operator 1> oder ... vor '<nom>'"
Entrez un opérateur valide à l'endroit indiqué.

4025

"Erwarte ':=' oder '=>' vor '<nom>'"
Entrez un des deux opérateurs à l'endroit indiqué.

4026

" 'BITADR' attend une adresse de bit ou une variable sur une adresse de bit. "
Utilisez une adresse de bit correcte (p.ex. %IX0.1).

4027

" Nombre entier ou constante symbolique attendu "
Insérez un nombre entier ou l'identificateur d'une constante correcte.

4028

" L'opérateur 'INI' requiert une instance de bloc fonctionnel ou une variable structurée "
Vérifiez le type des variables sur lesquelles vous appliquez l'opérateur INI.

4029

" Des appels imbriqués de la même fonction ne sont pas possibles."
Dans le cas de systèmes cibles non-réentrants et en mode simulation, un appel de fonction ne peut contenir comme paramètre un appel à lui-même.
Exemple : fun1(a,fun1(b,c,d),e);

Utilisez une variable intermédiaire.

4030

" 'ADR' requiert une variable comme paramètre et pas une expression ou une constante "
Remplacez la constante ou l'expression par une variable ou une adresse directe.

4031

" L'opérateur d'adresse n'est pas admis sur les bits! Veuillez utiliser 'BITADR'."
Utilisez BITADR. Veuillez noter : BITADR ne rend pas d'adresse mémoire physique.

4032

"' '<numéro>' d'opérandes sont trop peu pour '<nom>'. Un minimum de '<numéro>' est nécessaire "
Vérifiez combien d'opérandes sont requis par l'opérateur '<Nom>', et insérez les opérandes manquants.

4033

"' '<numéro>' opérandes sont trop pour '<nom>'. Le nombre exact de '<numéro>' est nécessaire "
Vérifiez combien d'opérandes sont requis par l'opérateur '<Nom>', et supprimez les opérandes superflus.

4034

" Division par '0'"
Vous utilisez une division par 0 dans une expression constante. Utilisez le cas échéant une variable avec la valeur 0 pour forcer une erreur d'exécution.

4040

" L'étiquette '<nom>' n'est pas définie "
Définissez une étiquette nommée <étiquette> ou bien remplacez <étiquette> par une étiquette définie.

4041

" Définition multiple de l'étiquette '<nom>' '"
L'étiquette de saut '<Nom>' est définie plusieurs fois dans le module. Renommez en conséquence ou enlevez une définition.

4042

" Il peut seulement être au maximum '<numéro>' renvoi de saut successivement "
Le nombre d'étiquettes par instruction est limité à '<nombre>'. Insérez une instruction fictive.

4043

"Le format du label est invalide. Un label doit être un nom qui peut être suivi par deux points. "
Soit le nom utilisé pour l'étiquette n'est pas un identificateur valide, soit il manque un signe deux-points dans la définition.

4050

" L'unité '<nom>' n'existe pas dans le projet "
Définissez un module nommé <Nom> au moyen de la commande de menu "**Projet**" + "**Insérer objet**" ou remplacez <Nom> par le nom d'un module défini.

4051

" '<nom>' n'est pas de fonction "
Utilisez pour le '<Nom>' un des noms de fonction définis dans le projet ou les bibliothèques.

4052

" '<nom>' doit être une instance déclarée du bloc fonctionnel '%s"
Utilisez pour le '<Nom d'instance>' une des instances de type '<Nom>' définies dans le projet ou modifiez le type de <Nom d'instance> en '<Nom>'.

4053

"%s' n'est pas de module ou opérateur valide"
Remplacez '<Nom>' par le nom d'un des modules définis dans le projet ou le nom d'un opérateur.

4054

"Bausteinname als Parameter von 'INDEXOF' erwartet"
Le paramètre indiqué n'est pas un nom de module valide.

4060

"'VAR_IN_OUT' Paramètre '<nom>' de '<nom>' requiert variable avec permission de modification comme entrée. "
Il faut transmettre aux paramètres VAR_IN_OUT des variables avec accès en écriture, car celles-ci peuvent être modifiées au sein du module.

4061

"'VAR_IN_OUT' Paramètre '<nom>' de '<nom>' doit être occupé. "
Les paramètres VAR_IN_OUT doivent être affectés à des variables avec accès en écriture, car celles-ci peuvent être modifiées au sein du module.

4062

"Pas d'accès au paramètre 'VAR_IN_OUT' '<nom>' de '<nom>' de l'extérieur."
Les paramètres VAR_IN_OUT ne peuvent être lus ou écrits qu'au sein du module car il s'agit d'un transmission par référence.

4063

"Vous ne pouvez pas attribuer des adresses de bit au paramètre 'VAR_IN_OUT' '<nom>' de '<nom>'. "
Une adresse de bit n'est pas une adresse physique correcte. Transmettez une variable ou une non-adresse directe de bit.

4064

"'VAR_IN_OUT' ne doit pas être écrasé dans un appel d'action local! "
Effacez l'assignation des variables VAR_IN_OUT pour l'appel local d'action.

4070

"Le module contient une expression trop complexe. "
Réduisez le niveau d'imbrication en divisant l'expression en plusieurs expressions à l'aide d'affectations à des variables intermédiaires.

4071

"Le réseau est trop grand "
Divisez le réseau en plusieurs réseaux.

4100

" '^' nécessite un type de pointer "
Vous essayez de déréférencer une variable qui n'est pas déclarée comme POINTER TO.

4110

" '[<index>]' seulement admis pour les variables de tableau "
Vous utilisez [<index>] pour une variable qui n'est pas déclarée comme ARRAY OF.

4111

"L'expression dans l'index d'un tableau doit avoir un résultat du type 'INT' "
Utilisez une expression de type correspondant ou une conversion de type.

4112

"Le tableau a trop d'indices "
Vérifiez le nombres d'indices (1, 2 ou 3) spécifiés dans la déclaration du tableau et supprimez les indices en surnombre.

4113

" Le tableau a trop peu d'indices "

Vérifiez le nombres d'indices (1, 2 ou 3) spécifiés dans la déclaration du tableau et complétez les indices manquants.

4114

" Il n'y a pas d'index constant dans la zone du tableau "
Assurez-vous que les indices utilisés se trouvent dans les limites du tableau.

4120

" Une variable structurée doit être placée devant '.' "
L'identificateur à gauche du point doit être une variable de type STRUCT ou un bloc fonctionnel FUNCTION_BLOCK ou encore le nom d'une FONCTION ou d'un PROGRAMME.

4121

"'<nom>' n'est pas de composant de '<nom>' "
Le composant '<Nom>' n'est pas contenu dans la définition de l'objet <Objet>.

4122

'<nom>' n'est pas de paramètre d'entrée du bloc fonctionnel appelé "
Vérifiez les variables d'entrée du bloc fonctionnel appelé et remplacez '<Nom>' par une de ces variables.

4200

" 'LD' attendu "
Insérez au moins une instruction LD dans la fenêtre d'éditeur du langage IL ou après l'étiquette.

4201

" Opérateur IL attendu "
Toute instruction IL doit débuter avec un opérateur ou une étiquette.

4202

" Fin inattendue de l'expression entre parenthèses "
Insérez une parenthèse droite (fermante).

4203

"'<nom>' ne doit pas être entre parenthèses "
L'opérateur indiqué n'est pas autorisé au sein d'une expression IL entre parenthèses.
(non autorisés : 'JMP', 'RET', 'CAL', 'LDN', 'LD', 'TIME')

4204

" Parenthèse de fermeture sans parenthèse d'ouverture correspondante "
Insérez une parenthèse gauche (ouvrante) ou effacez la parenthèse droite (fermante).

4205

" Une virgule est inadmissible après ')' "
Enlevez la virgule après la parenthèse droite (fermante).

4206

" Pas d'étiquettes dans des expressions entre parenthèses "
Déplacez l'étiquette en dehors de l'expression entre parenthèses.

4207

" Modificateur 'N' demande un opérande du type 'BOOL','BYTE','WORD' ou 'DWORD' "
Le modificateur 'N' nécessite un type de données pour lequel une négation booléenne peut être effectuée.

4208

" L'expression devant un opérateur doit livrer un résultat du type 'BOOL' "
Assurez-vous que l'expression donne un résultat booléen ou utilisez une conversion de type.

4209

" Pas de nom de fonction admis ici "
Remplacez l'appel de fonction par une variable ou par une constante.

4210

" 'CAL', 'CALC' ou 'CALN' requièrent une instance de bloc fonctionnel comme opérande "
Déclarez une instance du bloc fonctionnel que vous souhaitez appeler.

4211

" Commentaire dans IL seulement admis à la fin de la ligne "
Déplacez le commentaire à la fin de la ligne ou dans une ligne propre.

4212

" L'accumulateur est invalide devant un énoncé lié à la valeur de l'accu "
Le contenu de l'accumulateur n'est pas défini. Ceci se produit après des instructions qui ne donnent pas de résultat (p.ex. 'CAL').

4213

" 'S' et 'R' demande un opérande du type 'BOOL' "
Utilisez une variable booléenne à cet endroit.

4250

" Le module n'est pas terminée proprement: Insérez 'ST' ou effacez la dernière expression. "
La ligne ne commence pas avec une instruction ST valide.

4251

" La fonction '<nom>' a trop de paramètres "
Vous avez indiqué plus de paramètres que ce qui est déclaré dans la définition de la fonction.

4252

" La fonction '<nom>' a trop peu de paramètres "
Vous avez indiqué moins de paramètres que ce qui est déclaré dans la définition de la fonction.

4253

" 'IF' et 'ELSIF' demandent une condition de type booléen "
Assurez-vous que la condition suivant un 'IF' ou 'ELSEIF' soit une expression booléenne.

4254

" 'WHILE' requiert une condition booléenne"
Assurez-vous que la condition suivant un 'WHILE' soit une expression booléenne.

4255

" 'UNTIL' requiert une condition booléenne "
Assurez-vous que la condition suivant un 'UNTIL' soit une expression booléenne.

4256

" 'NOT' requiert un opérande booléen "
Assurez-vous que la condition suivant un 'NOT' soit une expression booléenne.

4257

" Le compteur de l'énoncé 'FOR' doit être du type 'INT' "
Assurez-vous que la variable de comptage soit un type de données entières (Integer) ou chaînes de bits (Bitstring) (p.ex. DINT, DWORD).

4258

" Le compteur dans l'énoncé 'FOR' n'est pas de variable avec permission de modification "
Remplacez la variable de comptage par une variable avec accès en écriture.

4259

" La valeur initiale de l'énoncé 'FOR' doit être du type 'INT' "
La valeur initiale de l'énoncé FOR doit être compatible avec le type de la variable de comptage.

4260

" La valeur finale de l'énoncé 'FOR' doit être du type 'INT' "
La valeur finale de l'énoncé FOR doit être compatible avec le type de la variable de comptage.

4261

" La valeur incrémentale de l'énoncé 'FOR' doit être du type 'INT' "
La valeur d'incrémentation de l'énoncé FOR doit être compatible avec le type de la variable de comptage.

4262

" 'EXIT' n'est admis que dans une boucle "
N'utilisez 'EXIT' qu'au sein d'énoncés 'FOR', 'WHILE' ou 'UNTIL'.

4263

" Nombre, 'ELSE' ou 'END_CASE' attendus"
Vous ne pouvez indiquer au sein d'un 'CASE' qu'un nombre ou un énoncé 'ELSE' ou encore l'énoncé final 'END_CASE'.

4264

" Le sélecteur de l'énoncé CACE doit être du type 'INT' "
Assurez-vous que le sélecteur soit un type de données entières (Integer) ou chaînes de bits (Bitstring) (p.ex. DINT, DWORD).

4265

" Après ',' nombre attendu "
Dans l'énumération des sélecteurs CASE, un autre sélecteur doit être placé après une virgule.

4266

" Un enoncé au minimum est demandé "
Entrez une instruction, au moins un point-virgule.

4269

" Après la branche 'ELSE', 'END_CASE' est attendu "
Fermez l'énoncé 'CASE' après la branche 'ELSE' au moyen de 'END_CASE'.

4270

" Constante de 'CASE' '%ld' déjà utilisé" "
Un sélecteur 'CASE' ne peut être utilisé qu'une fois dans un énoncé 'CASE'.

4271

" Le minimum indiqué est plus grand que le maximum. "
Corrigez les limites des sélecteurs de manière à ce que la limite inférieure ne soit pas plus grande que la limite supérieure.

4272

" Expectant paramètre '<nom>' au lieu de '<numéro>' dans l'appel de '<nom>'!"
Lorsque vous procédez au paramétrage des fonctions dans l'appel de fonctions avec le nom des paramètres, la position des paramètres (ordre) doit toujours et encore correspondre à la position des paramètres dans la définition des fonctions.

4300

"Saut ou Return requiert une entrée booléenne"
Assurez-vous que l'entrée pour un saut ou l'instruction de retour soit une expression booléenne.

4301

" Le module '<nom>' requiert exactement '<numéro>' entrées "
Le nombre d'entrées ne correspond pas au nombre de variables VAR_INPUT et VAR_IN_OUT indiqué dans la définition du module

4302

" Le module '<nom>' requiert exactement '<numéro>' sorties "
Le nombre de sorties ne correspond pas au nombre de variables VAR_OUTPUT indiqué dans la définition du module

4303

"'<nom>' n'est pas d'opérateur "
Remplacez '<Nom>' par un opérateur valide.

4320

" Expression non-booléenne '<nom>' avec contact "
Le signal de commutation d'un contact doit être une expression booléenne.

4321

" Expression non-booléenne '<nom>' avec bobinage "
La variable de sortie d'un bobinage doit être de type BOOL.

4330

"Une expression est attendue à l'entrée 'EN' de la boîte '<nom>'
Brancher l'entrée EN du module '<Nom>' sur une entrée ou une expression.

4331

" Une expression est attendue à l'entrée '<numéro>' de la boîte '<nom>' "
L'entrée du module d'opérateur n'est pas câblé.

4332

"Une expression est attendue à l'entrée '<nom>' du module '<nom>'"
L'entrée du module est du type VAR_IN_OUT et n'est pas câblé.

4333

" Identificateur attendu dans saut "
La destination du saut indiqué n'est pas un identificateur valide.

4334

" Une expression est attendue à l'entrée du saut "
Câblez l'entrée du saut avec une expression booléenne. Lorsque celle-ci a la valeur TRUE, le saut est exécuté.

4335

" Une expression est attendue à l'entrée de Return eturn"
Câblez l'entrée de l'instruction de retour avec une expression booléenne. Lorsque celle-ci a la valeur TRUE, le saut est exécuté.

4336

"Une expression est attendue à l'entrée de la sortie "
Reliez la sortie à une expression à laquelle cette sortie peut être affectée.

4337

" Identificateur attendu pour cette entrée "
Insérez dans la boîte d'entrée une expression ou un identificateur valide.

4338

" La boîte '<nom>' n'a pas d'entrées réelles "

Aucune des entrées du module d'opérateur '<Nom>' n'est câblé avec une expression valide. F

4339

"Types incompatibles à l'entrée: Ne peut pas convertir '<nom>' en '<nom>'"
L'expression dans la boîte de sortie n'est pas de type compatible avec l'expression qui devrait lui être affectée.

4340

"Saut requiert une entrée booléenne"
Assurez-vous que l'entrée pour le saut soit une expression booléenne.

4341

" Return requiert une entrée booléenne "
Assurez-vous que l'entrée pour l'instruction de retour soit une expression booléenne.

4342

" L'entrée 'EN' de la boîte requiert une entrée booléenne. "
Associez l'entrée EN du module avec une expression booléenne valide.

4343

" Affectation de la constante: %s
Vous avez déclaré l'entrée '<Nom>' du module '<Nom>' comme VAR_INPUT CONSTANT. Vous lui avez cependant affecté, dans la boîte de dialogue 'Editer paramètres', une expression qui n'est pas de type compatible.

4344

"'S' et 'R' demande un opérande du type 'BOOL' "
Insérez une expression booléenne valide derrière l'instruction Set (positionner) ou Reset (remettre à zéro).

4345

"Unzulässiger Typ für Parameter '<nom>' von '<nom>': Kann '<Typ>' nicht in '<Typ>' konvertieren."
Vous avez affecté à l'entrée '<Nom>' du module '<Nom>' une expression qui n'est pas de type compatible.

4346

" Une sortie ne doit pas être une constante. "
La cible d'une affectation doit être une variable ou une adresse directe avec accès en écriture.

4347

" Le paramètre 'VAR_IN_OUT' a besoin d'une variable avec accès écriture comme entrée. "
Il faut transmettre aux paramètres VAR_IN_OUT des variables avec accès en écriture, car celles-ci peuvent être modifiées au sein du module.

4350

"Une action de SFC ne peut pas être appelée de dehors! "
Des actions SFC ne peuvent être appelées qu'au sein du module SFC dans lequel elles ont été définies.

4351

" Le nom de l'étape n'est pas d'identificateur admissible: '<nom>' '"
Renommez l'étape et choisissez comme nom un identificateur valide.

4352

" Des caractères inadmissibles suivent le nom de l'étape admissible:'<nom>' '"
Effacez les caractères non autorisés dans le nom de l'étape.

4353

" Noms d'étape doubles: '<nom>' >'"
Renommez une des étapes.

4354

" Saut sur une étape non définie: '<nom>' '"
Choisissez comme destination du saut un nom d'étape disponible ou insérez une étape avec le nom non encore défini.

4355

"Eine Transition darf keine Seiteneffekte (Zuweisungen, FB-Aufrufe etc.) haben"
Une transition ne peut contenir qu'une expression booléenne.

4356

" Saut sans nom d'étape validé: '<nom>' "
Utilisez un identificateur valide comme destination de saut.

4357

" La bibliothèque CEI n'a pas été trouvée "
Vérifiez si la bibliothèque iecsfc.lib a été associée dans le gestionnaire de bibliothèques et si les chemins d'accès saisis dans les options de projet sont corrects.

4358

" Action pas déclarée: '<nom>' '"
Assurez-vous que l'action de l'étape CEI soit insérée dans l'Organisateur d'objets en dessous du module SFC et que le nom de l'action soit entré dans la case à droite du qualificateur.

4359

" Qualificateur interdit: '<nom>' "
Entrez un qualificateur pour l'action CEI dans la case située à gauche du nom de l'action.

4360

" Constante de temps attendue après qualificateur: '<nom>' >'"
Entrez une constante de temps pour l'action CEI dans la case située à gauche du nom de l'action, derrière le qualificateur.

4361

" '<nom>' n'est pas le nom d'une action "
Entrez une action définie dans le projet ou une variable booléenne pour l'action CEI dans la case située à droite du qualificateur.

4362

" Expression pas booléen dans l'action: '<nom>'"
Entrez une variable booléenne ou un nom d'action valide.

4363

" Le nom du pas CEI a déjà été utilisé pour une variable: '<nom>'"
Renommez soit l'étape, soit la variable.

4364

" Expression pas booléen dans une transition "
Le résultat de l'expression de transition doit être de type BOOLéenne.

4365

"L'etape '<nom>' a une valeur limite de temps fausse"
Ouvrez la boîte de dialogue des Attributs d'étape pour l'étape '<Nom>' et entrez une variable ou une constante de temps valide.

4366

" La marque pour l'étape parallèle n'est pas d'identificateur valide: '<nom>' '"
Entrez un identificateur autorisé à côté du triangle qui indique l'étiquette.

4367

" La marque '<nom>'existe déjà. "
Vous avez déjà désigné une étiquette ou une étape avec ce nom. Renommez-la en conséquence.

4368

" L'action '<nom>' est utilisée dans plusiers niveaux de SFC emboîtés! "
Vous utilisez l'action '<Nom>' dans le module et également dans une ou plusieurs actions de ce module.

4369

" Exactement un réseau nécessaire pour les transitions "
Vous avez utilisé pour la transition plusieurs réseaux FBD ou LD. Réduisez à un seul réseau.

4370

" Trouvé les lignes en trop après transition IL correcte "
Effacez les lignes non utilisées à la fin de la transition.

4371

" Des caractères inadmissibles suivent le nom de l'étape admissible:'<nom>' '"
Effacez les caractères non utilisés à la fin de la transition.

4400

" Le module '<nom>' est incomplet / a été importé ou converti en contenant des erreurs. "
Le module ne peut être entièrement converti en CEI 61131-3.

4401

" La constante de temps S5 est trop grand de %lu secondes (max. 9990s)."
L'accumulateur ne comporte pas de temps codé BCD valide.

4402

" L'accès direct n'est pas admis que sur les entrées et les sorties. "
Assurez-vous que vous n'accédez qu'à une variable définie comme entrée ou sortie.

4403

" Ordre STEP5 invalide ou pas convertible en CEI 61131-3. "
Toute commande STEP5 n'est pas convertible en CEI 61131-3, par exemple des commandes CPU comme MAS.

4404

" Opérande STEP5 invalide ou pas convertible en CEI 61131-3."
Toute opérande STEP5 n'est pas convertible en CEI 61131-3, ou il manque un opérande.

4405

"Le déverrouillage d'un timer STEP5/7 ne peut être converti en CEI 61131-3.."
Les temporisateurs CEI correspondants n'ont pas d'entrée de remise à zéro.

4406

" La constante de comptage STEP5 est trop grande (max. 999)"
L'accumulateur ne comporte pas de constante de comptage codée BCD valide.

4407

" L'instruction STEP5 n'est pas convertible en CEI 61131-3 "
Toute instruction STEP5/7 n'est pas convertible en CEI 61131-3, par exemple DUF.

4408

" L'accès de bit sur des mots timer/compteur n'est pas convertible en CEI 61131-3. "
Les commandes spéciales de temporisation / de comptage ne sont pas convertibles en CEI 61131-3.

4409

" Le contenu de Accu1 ou Accu2 n'est pas défini et ne peut être convertie en CEI 61131-3. "
Une commande qui associe deux accumulateurs n'est pas convertible car les contenus des accumulateurs sont inconnus.

4410

" Le module appelé n'existe pas dans le projet. "
Importez tout d'abord le module appelé.

4411

" Erreur dans la liste de variables globales. "
Veuillez vérifier le fichier SEQ.

4412

" Erreur interne No.11 "
Veuillez vous adresser au fabricant de l'automate.

4413

" Format erroné d'une ligne dans l'unité de données. "
Le code à importer contient une donnée erronée.

4414

" Nom FB/FX manque "
Il manque le nom symbolique d'un bloc fonctionnel (étendu) dans le fichier de sortie S5D.

4415

" Cette commande n'est pas permise après la fin du bloc "
Un module protégé ne peut être importé.

4416

" Commande invalide "
La commande S5 ne peut être désassemblée.

4417

" Commentaire pas terminé "
Terminez le commentaire avec "*)".

4418

" Nom FB/FX trop long (max. 8 caractères) "
Le nom symbolique d'un bloc fonctionnel (étendu) est trop long.

4419

"Format des lignes attendu ""(* Name: <FB/FX-Name> *)"" "
Corrigez la ligne en conséquence.

4420

"Nom du paramètre FB/FX manque"
Vérifiez les blocs fonctionnels.

4421

"Indication du type de paramètre FB/FX invalide"
Vérifiez les blocs fonctionnels.

4422

" Type de paramètre FB/FX pas indiqué"
Vérifiez les blocs fonctionnels.

4423

"Opérande actuel invalide "
Vérifiez l'interface du bloc fonctionnel.

4424

" Warning: Le bloc appelé n'existe pas ou le header est faux ou sans params. "
Le bloc fonctionnel appelé n'a pas encore été importé ou est incorrect ou ne possède pas de paramètres (vous pouvez dans ce dernier cas ignorer le message).

4425

" Etiquette pas définie "
La destination d'un saut n'est pas indiquée.

4426

" Le module n'a pas de nom STEP5 valide comme p.ex. PB10 "
Modifiez le nom du module.

4427

" Le type de timer n'a pas été indiqué "
Insérez une déclaration du temporisateur dans la liste des variables globales.

4428

" Nombre de parenthèses permis dans STEP 5 excédé "
Il n'est pas permis d'utiliser plus de sept parenthèses gauches (ouvrantes).

4429

" Erreur dans le nom formel du paramètre "
Le nom du paramètre ne peut comporter plus de quatre caractères.

4430

" Ce type de paramètre formel n'est pas convertible en CEI "
Des temporisateurs, compteurs et modules ne peuvent être convertis en tant que paramètres formels en CEI 61131-3.

4431

" Trop de paramètres 'VAR_OUTPUT' pour un appel dans STEP5-IL "
Un module ne peut compter comme sorties plus de seize paramètres formels.

4432

" Les étiquettes au milieu d'une expression sont interdites "
Les étiquettes ne peuvent pas être placées n'importe où dans CEI 61131-3.

4434

" Trop de labels "
Un module ne peut compter plus de 100 étiquettes.

4435

" Après saut/appel, vous ne pouvez plus continuer la liaison "
Après un saut ou un appel, il doit y avoir une commande de chargement.

4436

" Le contenu du résultat de liaison n'est pas défini et pas convertible en CEI 61131-3."
Une commande qui utilise le résultat de la liaison n’est pas convertible car la valeur du résultat de la liaison est inconnue.

4437

" Les types de la commande et de l'opérande ne correspondent pas "
Une commande bit à bit a été appliquée à un opérande de type mot ou vice-versa.

4438

" Pas de bloc de données ouvert (ajoutez 'A DB') "
Insérez un bloc de données A (A DB).

4500

" Variable ou adresse inconnue "
Cette variable d’espion n’est pas déclarée dans le projet. En appuyant sur la touche <F2> vous obtenez une liste de sélection pour l’édition relative à la variable déclarée.

4501

" Des caractères inadmissibles suivent une expression d'espion valide "
Supprimez les caractères en surnombre.

4520

" Erreur dans la directive pour le compilateur: Flag attendu devant '<nom>'! "
La pragma n’est pas saisie correctement. Vérifiez si '<Nom>' est un drapeau valide.

4521

" Erreur dans la directive pour le compilateur: Elément inattendu '<nom>'! "
Vérifiez la composition correcte de la pragma.

4522

" Directive 'flag off' attendue! "
La désactivation de la pragma manque, insérez une instruction 'flag off'.

4550

" Indice indéfinissé : Variable LdO '<numéro>', ligne '<numéro>' "
Assurez-vous que l’index soit situé dans la zone définie dans la configuration du système cible / Fonctions réseau.

4551

" Subindice indéfinissé : Variable LdO '<numéro>', ligne '<numéro>'"
Assurez-vous que le sous-index soit situé dans la zone définie dans la configuration du système cible / Fonctions réseau.

4552

"Indice indéfinissé : Parameter LdO '<numéro>', ligne '<numéro>'"
Assurez-vous que l’index soit situé dans la zone définie dans la configuration du système cible / Fonctions réseau.

4553

"Subindice indéfinissé : Parameter LdO '<numéro>', ligne '<numéro>'"
Assurez-vous que le sous-index soit situé dans la zone définie dans la configuration du système cible / Fonctions réseau.

4554

" Nome de variable invalide: Variable LdO '<numéro>', ligne '<numéro>' "
Entrez une variable de projet valide dans le champ Variable. Utilisez le format d'écriture <Nom du module> <Nom de la variable> ou <Nom de la variable> pour les variables globales.

4555

Insérez un valeur, n'est pas optional: Parameter OD '<numéro>', ligne '<numéro>'
Vous devez procéder à une saisie dans ce champ.

4556

" Insérez un valeur, n'est pas optional: Variable OD <nom>, ligne <nom>"
Vous devez procéder à une saisie dans ce champ.

10 Index

D

E

M

N

O

P

Q

R

S

T